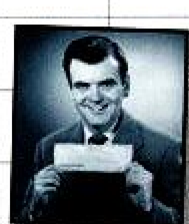
重视大脑的学习指南

Head First HTML与CSS、XHTML

（中文版）

轻松启动你的网页开发生涯



创建标准网页的初学者指南





注意常见的HTML与CSS陷阱和危险

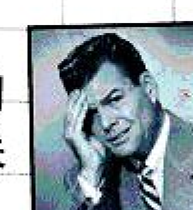


有关样式的种种误区

大约100个谜题与练习让你着迷



避免尴尬的验证错误



O'REILLY® 中国电力出版社

Elisabeth Freeman & Eric Freeman 著
林旺 张晓坤 译

Head First HTML与CSS、XHTML（中文版）

HTML/Web Programming

"每一章都是典雅的设计，每一章都透着实用与智慧。"

—*Ken Goldstein,*
Disney Online**执行副总裁**

"本书用非常现代的方式介绍网页标签以及呈现的实践技巧。"

—*Danny Goodman,*
《Dynamic HTML: The Definitive Guide》**作者**

"过去冗长的"试验－错误"学习过程被简化到这本平装书中！"

—*Mike Davidson,*
Newsvine**公司首席执行官**

"我喜爱《Head First HTML与CSS、XHTML》，它教给你快乐地学习HTML"

—*Sally Applin,*
用户界面设计师、艺术家

面对那些晦涩的HTML书你不禁要问："难道要成为专家之后才能读懂这些？"那么，你应该选择《Head First HTML与CSS、XHTML（中文版）》真正来学习HTML。这本书对你来说，将是一个系统学习创建工业标准Web页的体验，而不只简单地阅读：你将玩游戏、解决谜题、探索秘密，并以你从未想过的方式创建Web页。你还能学习HTML如何与CSS协同工作。当然，即使你没有听说过CSS，也没有什么大不了的——我们不会告诉别人你对CSS的认识还停留在1999年——如果你想在21世纪创建网页，你就必须知道和懂得CSS。

《Head First HTML与CSS、XHTML（中文版）》使你不再顾虑Web-safe颜色支持的浏览器问题，也不再不明智地在网页中加入<font>标签来控制字符。更好的是，你将兴致勃勃地学习HTML、XHTML和CSS，而不再昏昏欲睡。如果你曾经读过任何一本Head First系列书籍，你就会知道其中的奥秘：丰富、活泼的设计调动你的头脑，让思维活跃起来。通过采用神经生物学、认识心理学以及学习理论的最新研究成果，这本书将激发你学习HTML和CSS的兴趣。

学习创建网页的真实秘密，以及懂得你的老板在有关HTML表格认识方面的误区。更重要的是，在鸡尾酒会上，当你的同事偶然谈及他的HTML现在是如何严密、他的CSS是外部样式表时，你就应该坚持立场给予赞许，这会给酒会客人留下深刻的印象。



Elisabeth Freeman和Eric Freeman在推出《Head First设计模式》之后，开始在O'Reilly出版公司创作"Head First"系列丛书。之前，他们曾经在迪士尼公司从事互联网与数字媒体效果方面的工作，在他们的努力之下，有些网站已经成为业界的翘楚，包括ABCNews.com、Disney.com和ESPN.com。Elisabeth和Eric都是从耶鲁大学获得计算机学位；Elisabeth拥有理学硕士头衔，而Eric则是博士。

O'REILLY®
www.oreilly.com

O'Reilly Media, Inc.授权中国电力出版社出版

ISBN 978-7-5083-5646-4

9 787508 356464 >

定价：79.00元

此简体中文版仅限于在中华人民共和国境内（但不允许在中国香港、澳门特别行政区和中国台湾地区）销售发行

This Authorized Edition for sale only in the territory of People's Republic of China (excluding Hong Kong, Macao and Taiwan)

图书在版编目（CIP）数据

Head First HTML与CSS、XHTML（中文版）/（美）弗里曼（Freeman, E.）等著；林旺，张晓坤译，－北京：中国电力出版社，2008

书名原文：Head First HTML with CSS & XHTML

ISBN 978-7-5083-5646-4

I. H... II.①弗... ②林... ③张... III. ①超文本标记语言，HTML、XHTML－主页制作－程序设计 ②主页制作－软件工具，CSS IV. TP312 TP393.092

中国版本图书馆CIP数据核字（2007）第068184号

北京版权局著作权合同登记

图字：01-2007-1146号

©2005 by O'Reilly Media, Inc.

Simplified Chinese Edition, jointly published by O'Reilly Media, Inc. and China Electric Power Press, 2008. Authorized translation of the English edition, 2005 O'Reilly Media, Inc., the owner of all rights to publish and sell the same.

All rights reserved including the rights of reproduction in whole or in part in any form.

英文原版由O'Reilly Media, Inc.出版2005。

简体中文版由中国电力出版社出版，2008。英文原版的翻译得到O'Reilly Media, Inc.的授权。此简体中文版的出版和销售得到出版权和销售权的所有者 —— O'Reilly Media, Inc.的许可。

版权所有，未得书面许可，本书的任何部分和全部不得以任何形式重制。

书　　名/	Head First HTML与CSS、XHTML（中文版）
书　　号/	ISBN 978-7-5083-5646-4
责任编辑/	张旻
封面设计/	Ellie Volckhausen，Karen Montgomery，张健
出版发行/	中国电力出版社（www.infopower.com.cn）
地　　址/	北京三里河路6号（邮政编码100044）
印　　刷/	北京盛通印刷股份有限公司
开　　本/	787毫米×980毫米　16开本　43.5印张　968千字
版　　次/	2008年4月第1版　2008年4月第1次印刷
印　　数/	0001－4000册
定　　价/	79.00元（册）

敬告读者

本书封面贴有防伪标签，加热后中心图案消失。

本书如有印装质量问题，我社发行部负责退换。

版权所有　翻印必究

对《Head First HTML与CSS、XHTML》的赞誉

"《Head First HTML与CSS、XHTML》运用十分新颖的手法，介绍了网页标记及呈现方面的实践，具有很强的前瞻性。它准确地把握了读者的疑虑，并给予及时解决。丰富的图解及详实的步骤提供了最好的学习方式：只要做一点点改动，试着在浏览器上运行一下，就能够理解每个知识点的含义。"

——Danny Goodman，《Dynamic HTML：The Definitive Guide》作者

"Eric Freeman和Elisabeth Freeman十分清楚他们的目标是什么。随着因特网的日益复杂，富有创意的网页设计变得越来越关键。因此每一章的核心就是精美的设计，每一个概念的阐述都具有实用性，却又不失趣味性。"

——Ken Goldstein，迪士尼在线执行副总裁兼总经理

"如果每个HTML设计人员都从读这本书开始，那么，Web世界将变得更加迷人。"

——L.David Baron，Mozilla 网页布局及CSS技术主管
http:// bdaron.org/

"迄今为止，我编写HTML和CSS已经有十年了。通过这部神奇的书籍，过去所需要的长期摸索和反复试验，如今被巧妙地规避了。HTML曾经只要能正确显示就万事大吉，但是随着网络标准的出现，对网站可访问性的要求越来越高，人们已不能再忍受草率的编码了……无论从商业角度，还是从社会责任角度来看，都是这样的。《Head First HTML与CSS、XHTML》教你如何从一开始就正确地做事，而不使整个过程显得无所适从。应该这样说，通过Eric和Elisabeth出色的工作，HTML并不比日常的语言复杂，每个概念都变得一目了然。"

——Mike Davidson，Newsvine,Inc总裁兼CEO

"这本书所涵盖的信息同教授知道的一样，但是，通过一种幽默易懂的方式来诠释你就不会觉得这些东西难以掌握。"

——Christopher Schmitt，《The CSS Cookbook》和《Professional CSS》作者
schmitt@christopher.org

"你们编写了一本通俗易懂的XHTML的书，就连一个CEO都能读懂。接下来你们要写什么?是不是简单得连我的开发人员都能明白？至于下一步，我们将合作组建一个团队或进行其他方式的合作。"

——Janice Fraser，Adaptive Path CEO

更多对《Head First HTML与CSS、XHTML》的赞誉

"我心中的《Head First HTML与CSS、XHTML》——让你轻松愉快地学到你想学的东西。"

—— Sally Applin, UI设计者和艺术家, http://sally.com

"这本书幽默而又魅力十足，最重要的是，它有一颗为读者服务的心。这样评价一本关于技术的书有点荒谬，但我确实觉得这本书（至少是作者）的核心在于关心读者是否学到了东西。这在本书的风格、语言和技术上都有所体现。Eric和Elisabeth知道——确切地说是了解——读者的需要。这里要感谢你为本书明确地遵循标准所作出的努力。很高兴能看到这样一本入门级的书，我想它将得到广泛地阅读和学习——这种通俗易懂的普及活动也代表了遵循网页编码标准的价值。我甚至从这里发现了一些以前没有考虑过的论据——当我被问到'标准支持有什么用？为什么我们要关注这个问题？'时，我可以用这本书作为佐证。现在我有了更多的论据！我喜欢这本书还因为它告诉我们网页工作的机制——FTP、Web服务器基础知识及文件结构等。"

—— Robert Neer, Movies.com产品研发经理

"这本《Head First HTML与CSS、XHTML》是教你如何制作出优秀网页的最有趣的书。它不仅涵盖了HTML、CSS和XHTML的全部内容，而且，通过大量优秀的示例诠释外行不懂的术语。我发现读这本书真是一种享受，并且能从中学到新的东西。"

—— Newton Lee, "ACM Computers in Entertainment"主编
http://www.acmcie.org

"我的妻子偷走了我的这本书。因为她从来没有接触过网页设计，所以，她需要一本类似《Head First HTML与CSS、XHTML》这样的书，帮助她由浅入深地掌握网页设计。现在，她有一堆想做的网站——我儿子的班级、我们的家庭……如果我足够幸运的话，当她做完这些网站后，我才能够拿回这本书。"

—— David Kaminsky, IBM发明家

"记住，如果你习惯于晚上用读书来催眠，那么就只能在白天阅读《Head First HTML与CSS、XHTML》，因为它会使你大脑保持清醒。"

—— Pauline McNamara, 瑞典Fribourg大学新技术和教育中心

对作者著作的早期赞誉

继重磅推出的《Head First Java》后，这本书采用各种各样的手段帮助你理解和记忆。它不仅增加了很多图片：引起人们兴趣的人物图片，还有令人惊喜不断的是小故事，因为人们都乐于听故事（诸如比萨饼和巧克力之类的故事）。此外，它确实很有趣。

—— Bill Camarda, READ ONLY

“这本书结构清晰，内容详实，语言幽默，妙趣横生。就算是非程序员，也能用它迅速解决问题。”

—— Cory Doctorow, Boing Boing联合主编
《Down and Out in the Magic Kingdom》
和《Someone Comes to Town,Someone Leaves Town》作者

“我感觉读这本书的效果相当于读一千磅同类书的效果。”

—— Ward Cunningham, Wiki的发明者、Hillside Group创始人

“这本书近乎完美，它既介绍了专业知识，又有很强的可读性。叙述权威，文笔优美。这是我读过的让我觉得不可或缺的几本软件书之一（我大约摒弃了十本这方面的书）。”

—— David Gelernter，耶鲁大学计算机科学系教授，《Mirror Worlds》、《Machine Beauty》作者

“完全引领你深入到一种模式中，在这里可能将复杂的事情变简单，但也可能将简单的事情变复杂。我认为最好的向导莫过于Freemans。

—— Miko Matsumura, Middleware行业分析师，
Sun Microsystem前首席Java布道者（Evangelist）

“我为它哭，我为它笑，它深深地感动了我。”

—— Daniel Steinberg,java.net主编

“就像我们当中一位随和的编码高手，对那些实用的开发策略娓娓道来，既让我们轻松掌握，又不会被老学究般的陈词滥调搞得身心疲惫。”

—— Travis Kalanick,Scour and Red Swoosh创始人
麻省理工学院(MIT)年度全球青年技术杰出人物100名(TR100)

“我实在喜欢这本书。事实上，我当着我妻子的面亲吻了它。”

—— Satish Kumar

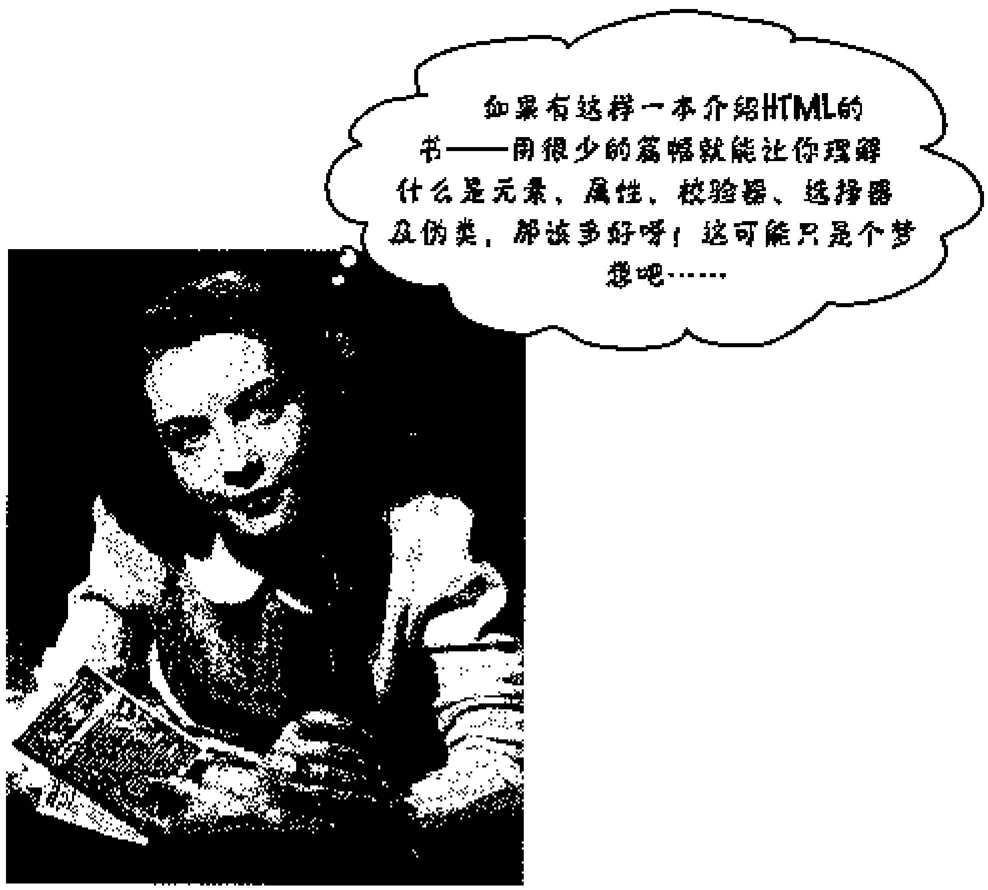
O'Reilly的其他相关书籍

HTML Pocket Reference
CSS Pocket Reference
CSS Cookbook
Cascading Style Sheets:The Definitive Guide
HTML & XHTML: The Definitive Guide
Dynamic HTML:The Definitive Reference
Learning Web Design:A Beginner's Guide to HTML,Graphics and Beyond

O'Reilly的其他Head First系列书籍

Head First Design Patterns
Head First Java
Head First Serlvets and JSP
Head First EJB
Head First PHP & MySQL
Head First JavaScript
Head Rush Ajax

Head First HTML与CSS、XHTML（中文版）



Elisabeth Freeman

Eric Freeman 著

林旺 张晓坤 译

O'REILLY®

Beijing • Cambridge • Köln • Paris • Sebastopol • Taipei • Tokyo

O'Reilly Media, Inc.授权中国电力出版社出版

中国电力出版社

O’Reilly Media, Inc.介绍

为了满足读者对网络和软件技术知识的迫切需求，世界著名计算机图书出版机构O’Reilly Media, Inc.授权中国电力出版社，翻译出版一批该公司久负盛名的英文经典技术专著。

O’Reilly Media, Inc.是世界上在 Unix、X、Internet 和其他开放系统图书领域具有领导地位的出版公司，同时也是联机出版的先锋。

从最畅销的 *The Whole Internet User’s Guide & Catalog*（被纽约公共图书馆评为20世纪最重要的50本书之一）到GNN（最早的Internet门户和商业网站），再到WebSite（第一个桌面PC的Web服务器软件），O’Reilly Media, Inc.一直处于Internet发展的最前沿。

许多书店的反馈表明，O’Reilly Media, Inc.是最稳定的计算机图书出版商——每一本书都一版再版。与大多数计算机图书出版商相比，O’Reilly Media, Inc.具有深厚的计算机专业背景，这使得O’Reilly Media, Inc.形成了一个非常不同于其他出版商的出版方针。O’Reilly Media, Inc.所有的编辑人员以前都是程序员，或者是顶尖级的技术专家。O’Reilly Media, Inc.还有许多固定的作者群体——他们本身是相关领域的技术专家、咨询专家，而现在编写著作，O’Reilly Media, Inc.依靠他们及时地推出图书。因为O’Reilly Media, Inc.紧密地与计算机业界联系着，所以O’Reilly Media, Inc.知道市场上真正需要什么图书。

《Head First HTML与CSS、XHTML》的作者





Elisabeth是作者、软件开发人员及数字艺术家。她很早就开始进行Internet相关的研究，也是Ada Project的共同发起人（Ada Project是一个针对在计算机界工作的女性而设计的网站，曾获得大奖，现在已经并入ACM）。最近她带领迪士尼的数字媒体研发力量与他人共同发明了一个名为Motion的内容系统，此系统每天传送巨量的数字内容给迪士尼、ESPN及Movies.com的用户。

Elisabeth本质上是一个计算机科学家，拥有耶鲁大学和印第安那大学的计算机科学硕士学位。她的工作领域很广，包括视觉语言、RSS内容整合与Internet系统。她也很积极提倡女性从事计算机工作。今天，你可以发现她在她的Mac上使用Java或Cocoa，但是其实，她最希望的是全世界都使用Scheme。从小在苏格兰长大，Elisabeth喜欢在大自然踏青及户外活动。一旦她在户外，相机总是不离手。她热爱骑单车，是个素食主义者，也很喜欢动物。

她的电子邮件信箱是beth@wickedlysmart.com，你可以发电子邮件给她。

Eric是一个计算机科学家，热衷于软件架构和媒体。

他刚刚花了四年的时间在一个梦寐以求的工作上：在迪士尼指导Internet宽带与无线应用。现在，他回到写作的岗位上，用Java和Mac创造很酷的软件。

在90年代，Eric和David Gelernter一起花了大量的时间，寻找Desktop metaphor的替代品。（他们“仍然”在问：我干嘛不得不给计算机文件取个名字）。也因为这样的研究，Eric在1997年获得耶鲁大学的博士学位。他也与他人一同创立了Mirror Worlds Technologies公司（已经被收购），将他的论文内容商业化，创建了一套软件Lifestreams。

以前，Eric为网络和超级计算机写软件，你可能通过《JavaSpaces Principles Patterns and Practice》这本书得知他的名号。他曾在Thinking Machine CM-5上实现了元组空间系统（tuple-space system），也在80年代末期为NASA创建了第一个Internet信息系统，他为此深感自豪。

Eric目前住在圣达菲附近的沙漠中，当他不写书或代码时，他总是花更多时间摆弄他的家庭影院，而不是观看影片，他利用空档时间试着修复80年代的经典视频游戏Dragon Lair。他也不介意在晚上兼差当个电音DJ。

给他的E-mail可以写到eric@wickedlysmart.com，你也可以去参观他的Blog，网址在http://www.ericfreeman.com。

如何使用这本书

引子



有一个问题真是听得我们耳朵都磨出茧了，这就是："你们为什么要把这样一些东西放在一本HTML书里呢？"这一节正是要回答这个问题。

谁适合读这本书？

如果对下面的所有问题你都能肯定地回答“是”：

① 你是否通过Web浏览器和文本编辑器来访问计算机？

如果你能使用过去十年中制造的计算机，回答就是肯定的。

② 你想学习、理解并记住如何使用最好的技术和最新的标准制作网页吗？

③ 你是不是更喜欢一种轻松的氛围，就像在餐桌上交谈一样，而不愿意被动地听技术报告似的枯燥乏味的说教？

那么，这本书正是你需要的。

谁暂时还不适合读这本书？

如果满足下面任何一种情况：

① 你是不是对计算机一无所知？

（你不需要有很深入的研究，但至少应该了解文件夹和文件、简单的文本编辑软件，以及如何使用Web浏览器。）

② 你是不是一个很棒的Web开发人员，正在找一本参考书？

③ 你是不是对新鲜事物都畏手畏脚？你是不是宁愿接受牙根管治疗，也不愿意接受苏格兰花格裙？你是不是觉得，如果把HTML标记都拟人化了，这样的一本书肯定不是一本正儿八经的技术书？



那么，太遗憾了，本书不适合你。

［营销备注：本书适合所有有信用卡的人。］

我们知道你在想什么

“这算一本正儿八经的编程书吗？”

“这一堆图是干什么的？”

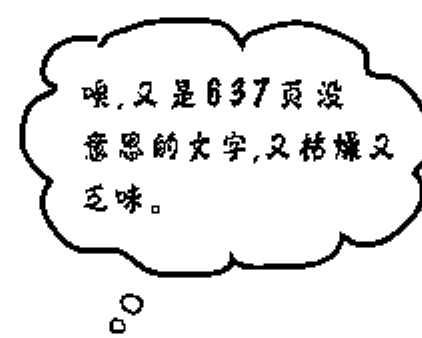
“我真的能这样学吗？”



我们也知道你的大脑在想什么。

你的大脑总是渴求一些新奇的东西，它一直在搜寻、审视、期待着不寻常的事情发生。大脑的构造就是如此。正是这一点才让我们不至于固步自封，能跟着时代前进。

如今，一般是不太可能被老虎吃掉的。然而，你的大脑还是一直在注意着周围是否有潜伏的老虎。只不过你自己没有意识到而已。但是我们每天都会遇到许多按部就班的事情，这些事情很普通，对于这样一些例行的事情或者平常的东西，你的大脑又是怎么处理的呢？它的做法很简单，就是不让这些平常的东西妨碍大脑真正的工作。那么什么是大脑真正的工作呢？这就是记住那些确实重要的事情。它不会费心地去记乏味的东西；就好像大脑里有一个筛子，这个筛子会筛掉“显然不重要”的东西，如果遇到的事情枯燥乏味，这些东西就无法通过这个筛子。



那么你的大脑怎么知道到底哪些东西重要呢？打个比方，假如你某一天外出旅行，突然一只大老虎跳到你面前，此时此刻，你的大脑里会发生什么呢？

看到这只大老虎，你的神经元会“点火”，情绪爆发，释放出一些化学物质。

好了，这样你的大脑就会知道……

这肯定很重要！可不能忘记了！



不过，假如你正待在家里或者坐在图书馆里，这里很安全、很温暖，肯定没有老虎。你正在刻苦学习，准备应付考试。也可能想学一些比较难的技术，你的老板认为掌握这种技术需要一周时间，最多不超过十天。这就存在一个问题。你的大脑很想给你帮忙。它会努力地把这些显然不太重要的内容赶走，保证这些东西不去侵占本不算充足的脑力资源。这些资源最好还是用来记住确实重要的事情，比如大老虎，再比如火灾险情。如果你曾经只身着短衣裤被大雪围困，这件事肯定不会忘却，你的大脑会记住绝不要让这种情况再发生第二次。

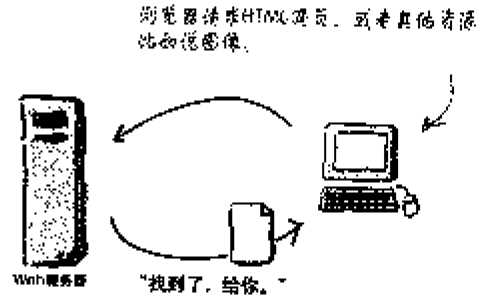
我们没有一种简单的方法来告诉大脑：“嘿，大脑，真是谢谢你了，不过不管这本书多没意思，也不管我对它是多么的无动于衷，但我确实希望你能帮助我把这些东西记下来。”

我们认为，"Head First"的读者就是要学习的人。

那么，怎么来学习呢?首先，必须了解，然后要保证自己确实不会忘记。这可不是填鸭式的硬塞。根据认知科学、神经生物学和教育心理学的最新研究，学习途径相当丰富，绝非只是通过书本上的文字。我们很清楚怎么让你的大脑兴奋起来。

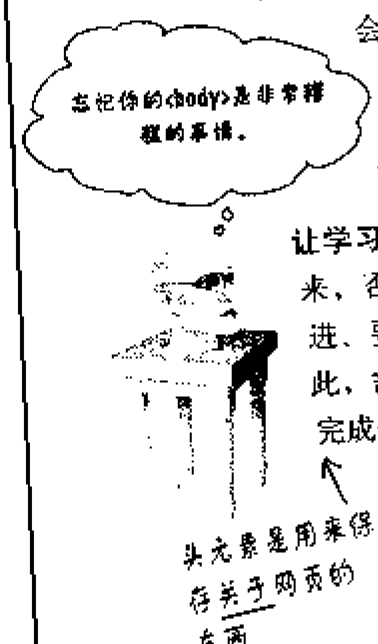
下面是一些Head First学习原则：

看得到。与单纯的文字相比，图片更能让人记得住，通过图片，学习效率会更高（对于记忆和传输型的学习，甚至能有多达89%的效率提升）。而且图片更能让人看懂。

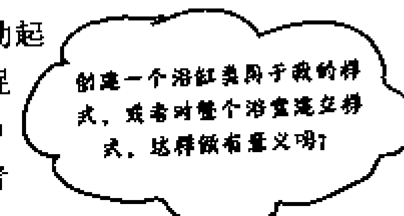


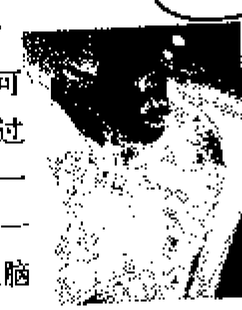
把文字放在与之相关的图片内部或周围，而不是把图片放在一页的最下面或另一页上，学习者的能力就能得到多至两倍的提高，从而能更好地解决有关的问题。

采用一种针对个人的交谈式风格。最新的研究表明，如果学习过程中采用一种第一人称的交谈方式直接向学生讲述有关内容，而不是用一种干巴巴的语调介绍，学生在学习之后的考试中成绩会提高40%。正确的做法是讲故事，而不是做报告。要用通俗的语言。另外不要太严肃。如果你面对着这样两个人，一个是你在餐会上结识的很有意思的朋友，而另一个学究气十足，喋喋不休地对你说教，在这两个人中，你会更注意哪一个呢?



让学习的人想得更深。换句话说，除非你很积极地让神经元活动起来，否则你的头脑里什么也不会发生。必须引起读者的好奇，促进、要求并鼓励读者去解决问题、得出结论、产生新的知识。为此，需要提出挑战，留下练习题和拓宽思路的问题，并要求读者完成一些实践活动，让左右脑都开动起来，而且要利用多种思维。





引起读者的注意，而且要让他一直保持注意力。我们可能都有这样的体验，"我真的想把这个学会，不过看过一页后就变得昏昏欲睡了"。你的大脑注意的是那些不一般、有意思、有些奇怪、抢眼的、意料之外的东西。学习一项有难度的新技术并不一定枯燥。如果学习过程不乏味，你的大脑很快就能学会。

影响读者的情绪。现在我们知道了，记忆能力很大程度上取决于所记的内容对我们的情绪有怎样的影响。如果是你关心的东西，就肯定记得住。如果让你感受到了什么，这些东西就会留在你的脑海中。不过，我们所说的可不是什么关于男孩与狗的伤心故事。这里所说的情绪是惊讶、好奇、觉得有趣。想知道"什么……"，还有就是一种自豪感，如果你解决了一个难题，学会了所有人都觉得很难的东西，或者发现你了解的一些知识竟是那些自以为无所不能的傲慢家伙所不知道的，此时就会有一种自豪感油然而生。



元认知：有关思考的思考

如果你是真的想学，而且想学得更快、更深入，就应该注意你怎样才能集中注意力。考虑自己是怎样思考的，并了解自己的学习方法。

我们中间大多数人长这么大可能都没有上过有关元认知或学习理论的课程。我们想学习，但是很少有人教我们怎么来学习。



不过，这里可以做一个假设，如果你手上有这本书，你想学创建网页，而且可能不想花太多时间。另外，你想记住你读到的所有内容，并且能够运用自如。为此必须理解这些内容。想要最大程度地掌握这本书或其他任何一本书中介绍的知识，就要让你的大脑负起责任来，要求它记住这些内容。

怎么做到呢？技巧就在于要让你的大脑认为你在学习的新东西确实很重要，对你的生活有很大影响。就像老虎出现在面前一样。如若不然，你将陷入旷日持久的拉锯战中，虽然你很想记住所学的新内容，但是你的大脑却会竭尽全力地把它们拒之门外。

那么，究竟怎样才能让你的大脑把HTML & CSS看作是一只饥饿的老虎呢？

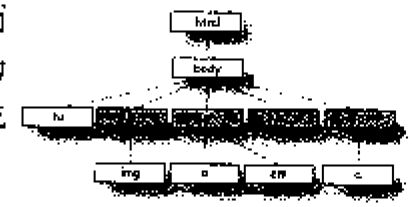
这有两条路：一条比较慢，很乏味；另一条路不仅更快，还更有效。慢方法就是大量地重复。你肯定知道，如果反反复复地看到同一个东西，即使再没有意思，你也能学会并记住它。如果做了足够的重复，你的大脑就会说“尽管看上去这对他来说好像不重要，不过，既然他这样一而再、再而三地看同一个东西，那么我就假定这是很重要的吧。”

更快的方法是尽一切可能让大脑活动起来，特别是开动大脑来完成不同类型的活动。如何做到这一点呢？上一页列出的学习原则正是一些主要的可取做法，而且经证实，它们确实有助于让你的大脑全力以赴。例如，研究表明，把文字放在所描述图片的中间（而不是放在这一页的别处，比如作为标题，或者放在正文中），这样会让你的大脑更多地考虑这些文字与图片之间有什么关系，而这就会让更多的神经元点火。让更多的神经元点火＝你的大脑更有可能认为这些内容值得注意，而且很可能需要记下来。

交谈式风格也很有帮助，当人们意识到自己在与“别人”交谈，往往会更加关注，这是因为他们总想跟上谈话的思路，并能做出适当的发言。让人惊奇的是，大脑并不关心“交谈”的对方究竟是谁，即使你只是与一本书“交谈”，它也不会在乎！另一方面，如果写作风格很正式，干巴巴的，你的大脑就会觉得像坐在一群人当中被动地听人做报告一样，很没意思，所以不必在意对方说的是什么，甚至可以打瞌睡。

不过，图片和交谈式风格还只是开始而已，能做的还有很多。

我们是这么做的：



我们用了很多图片，因为你的大脑更能接受看得见的东西，而不是纯文字。对你的大脑而言，一幅图顶得上1024个字。如果既有图片又有文字，我们会把文字放在图片当中，因为文字处在所描述的图片中间时，大脑的工作效率更高，倘若把这些描述文字作为标题，或者“湮没”在别处的大段文字中，那就达不到这种效果了。

我们采用了重复手法，会用不同的方式，采用不同类型的媒体、运用多种思维手段来介绍同一个东西，目的是让有关内容更有可能储存在你的大脑中，而且能够在多个区中都有容身之地。



扮演浏览器

我们会用你想不到的方式运用概念和图片，因为你的大脑喜欢新鲜玩艺儿，在提供图片和想法时，至少会带有一些情绪因素，因为如果能产生情绪反应，你的大脑就会投入更大的注意。而这会让你感觉到这些东西更有可能要被记住，其实这种感觉可能只是有点幽默、让人奇怪或者比较感兴趣而已。

我们采用了一种针对个人的交谈式风格，因为当你的大脑认为你在参与一个交谈，而不是被动地听一场演示汇报时，它就会更加关注。即使你实际上在读一本书，也就是说在与书“交谈”，而不是真正与人交谈，但这对你的大脑来说并没有什么分别。

在这本书里，我们加入了100多个实践活动，因为与单纯的阅读相比，如果能实际做点什么，你的大脑会更乐于学习，更愿意去记。练习都是我们精心设计的，有一定的难度，但是确实能做出来，因为这是大多数人所希望的。



要点

我们采用了多种学习模式，因为尽管你可能想循序渐进地学习，但是其他人可能希望先对整体有一个全面认识，另外可能还有人只是想看一个代码示例。不过，不管你想怎么学，要是同样的内容能以多种方式来表述，这对每一个人都会有好处。

谜题



这里的内容不只是单单涉及左脑，也不只是让右脑有所动作，我们会让你的左右脑都开动起来，因为你的大脑参与得越多，你就越有可能学会并记住，而且能更长时间地保持注意力。如果只有一半大脑在工作，通常意味着另一半有机会休息，这样你就能更有效率地学习更长时间。

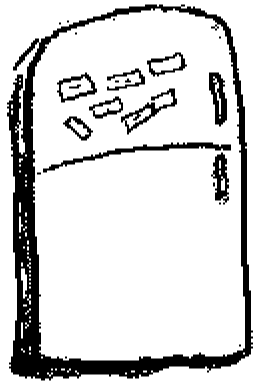
我们会讲故事，留练习，从多种不同的角度来看同一个问题，因为如果要求大脑做一些评价和判断，它就能更深入地学习。



你会看到我们给出的一些练习，还要回答一些问题，这些问题往往不是直截了当就能做出回答的，通过克服这些挑战，你就能学得更好，因为让大脑真正做点什么的话，它就更能学会并记住。想想吧，如果只是在健身馆里看着别人流汗，这对于保持你自己的体形肯定不会有什么帮助，正所谓临渊羡鱼，不如退而结网。不过另一方面，我们会竭尽所能不让你钻牛角尖，把劲用错了地方，而是能把功夫用在点子上。也就是说，你不会为搞定一个难懂的例子而耽搁，也不会花太多时间去弄明白一段晦涩难懂而且通篇行话的文字，我们的描述也不会太过简洁而让人无从下手。

我们用了拟人手法。在故事中，在示例中，还有在图片中，你都会看到人的出现。这是因为你本身是一个人，不错，这就是原因。如果和人打交道，相对于东西而言，你的大脑会更集中注意力。

我们充分利用了80/20方法，我们认为，如果你真的要成为一个很棒的Web开发人员的话，那么这肯定不会是你唯一的书，所以我们不打算面面俱到。这里只提供了你真正需要的东西。



沿着虚线剪下，贴在冰箱上。

可以用下面的方法让你的大脑就范

好了，我们该做的已经做了，剩下的就要看你自己的了。这些提示只是个开头：听一听你的大脑是怎么说的，弄清楚对你来说哪些做法可行，哪些做法不能奏效。还可以做些新的尝试。

---

① 慢一点，你理解的越多，需要记的就越少。

不要光是看看而已。停下来，好好想一想。书中提出问题的时候，你不要直接去翻答案。可以假想成真的有人在问你问题。你让大脑想得越深，就越有可能学会并记住。

② 勤做练习，自己记笔记。

我们给你留了练习，但是如果这些练习的解答也由我们一手包办，那和有人替你参加考试有什么区别？不要只是坐在那里看着练习发呆。拿出笔来，写一写、画一画。大量研究都证实，学习过程中如果能实际动动手，将改善你的学习效果。

③ 阅读“There are no Dumb Questions”部分。

顾名思义，这些问题可不是可有可无的旁注，它们绝对是核心内容的一部分！千万不要把它们跳过去不看。

④ 上床睡觉之前不要再看别的书了，或者至少不再看其他有难度的东西。

学习中有一部分是在你合上书之后完成的（特别是，要把学到的知识长久地记住，这往往无法在看书的过程中做到）。你的大脑也需要有自己的时间来再做一些处理。如果在这段处理时间内你又往大脑里灌输了新的知识，那么你刚学的一些东西就会被丢掉。

⑤ 要喝水，而且要多喝点水。

如果能提供充足的液体，你的大脑才能有最佳表现。如果缺水（可能你觉到口渴之前，就已经缺水了），学习能力就会下降。

⑥ 大声说出来。

说话可以刺激大脑的另一部分。如果你想看懂什么，或者想更牢地记住它，就要大声说出来。更好的办法是，大声地解释给别人听。这样你会学得更快，而且可能会有一些新的认识，而这是以前光看不说的时候未曾发现的。

⑦ 听听你的大脑怎么说。

注意一下你的大脑是不是负荷太重了，如果发现自己开始浮光掠影地翻看，或者刚看的东西就忘记了，这说明你该休息一会儿了。达到某个临界点时，如果还一味地向大脑里塞，这对加快学习速度根本没有帮助，甚至还可能影响正常的学习。

⑧ 要有点感觉！

你的大脑需要知道这是很重要的东西。要真正融入到书中的故事里。为书里照片加上你自己的说明。你可能觉得一个笑话很憋脚，不太让人满意，但这总比根本无动于衷要好。

⑨ 设计一些东西！

把这应用到你最近设计的网页或者旧项目上。干些事情，以获取书上练习和其他项目以外的经验。你所要做的只是准备一支笔和一个等待解决的问题……嗯，一个对你使用HTML和CSS有帮助的问题。

Readme

这是一本介绍关于学习经验的书而不是参考书，我们有意删去了许多无关的内容。第一次阅读时，你需要从头开始，因为本书对读者的背景知识已经做好假设了。

我们从讲授基础HTML开始，然后是基于标准的HTML4.01，最后是XHTML。

要编写基于标准的HTML或者XHTML，有许多技术性细节你需要理解，尽管这些在你试图学习HTML基础时不起作用。我们的方案是先让你学习基本的HTML概念（而不必管这些细节），当你对HTML有个抽象的认识时，再教你如何编写基于标准的HTML和XHTML。这样做的好处是在你学习完基础后，这些技术细节就开始发挥作用了。

在你开始使用CSS时，编写合乎规范的HTML或XHTML也很重要，所以在你接受关于CSS的困难工作前我们先要重点让你弄懂基于标准的HTML和XHTML。

我们不可能涉及所有独立创造出来的HTML元素、属性或CSS特性。

各种HTML元素、属性、CSS特性数不胜数，虽然它们很有趣，但我们的目标是写一本通俗易懂而又不太厚的书，所以在这里我们就不把它们一一列出了。我们关注的是对你这个初学者有用的关键的HTML元素和CSS特性，并确保你能够真正地、深入地彻底了解如何使用它们，以及何时使用它们。我们相信，读过本书之后，你可以很快地从其他资源中学到本书没有介绍的元素和属性，并且游刃有余。

这本书提倡把你的网页结构和呈现彻底分离。

今天，许多网站使用HTML和XHTML来组织内容，并使用CSS来做样式和呈现。20世纪90年代的网页经常使用不同的模型，HTML不但用于组织，也用于样式。本书教你如何使用HTML来组织，使用CSS来做样式；我们认为没有必要教你过时的坏习惯。

在本书中，我们鼓励你使用多种浏览器。

学习写基于标准的HTML、CSS和XHTML时，你会（也许是经常）发现在用不同的浏览器浏览网页时会有细微的差别。所以，我们鼓励你使用最少两种目前流行的浏览器来测试你的网页。观察不同浏览器中的差别，编写在多种浏览器中都能顺利运行的网页，这些都会令你受益无穷。

我们经常使用标记名作为元素名。

我们使用标记名“<a>元素”，而不是“a元素”或者“‘a’元素”。尽管这并不是技术原因（因为<a>是一个开始标记，而不是完全展开的元素），但这会令程序更具可读性，而且我们经常在后边加上“元素”两字以免混淆。

实践活动不是随便安排的。

练习和实践活动不是随便安排的；它们是本书核心内容的重要组成部分。有的有助于记忆，有的有助于理解，有的有助于指导你如何学以致用。不要忽略练习。虽然你要做的只不过是些纵横字谜，但它们会让你的大脑把这些词和上下文联系起来，从而达到记忆的效果。

我们有意安排了许多重复，这些重复非常重要。

Head First系列图书有一个与众不同的地方，这就是，我们希望你确确实实地掌握这些知识。另外，我们希望在学完这本书之后你能记住学过了什么。尽管重复很有必要，不过，多数参考书都不认为重复和回顾是一个重要的环节，但是在这本书里，你会看到一些概念会一而再、再而三地出现很多次。

代码示例尽可能短小精焊。

有读者告诉我们，如果在一个示例中查了200行才能找到要理解的那两行，这是很让人郁闷的。这本书里大多数示例往往都开门见山，作为上下文的代码会尽可能地少，这样你就能一目了然地看到哪些东西是你需要学习的。别指望这些代码很健壮，要知道这里的代码甚至是不完整的——毕竟我们的代码是辅助学习之用，所以不见得一定功能完整。

我们已经将所有的示例文件都放在网络上，可供下载。网址是：**http://www.headfirstlabs.com/books/hfhtml/**。

“Brain Power”习题没有答案。

对于某些人来说，“Brain Power”习题没有对的答案，对于另一些人来说，“Brain Power”习题所带来的学习经验在于决定你的答案是否是对的，以及何时你的答案是对的。在某些“Brain Power”习题中，我们会提供暗示，为你指引正确的方向。

技术审校群

Joe Konior

Valentin Crettaz

Corey McGlone





Louise Barr





Barney Marispini



埃菲尔铁塔

本书狂热的审校团队无畏的队长

Marcus Green







Johannes de Jong

Pauline McNamara

Pauline得到了"最严厉审校"奖。

Ike Van Atta



David O'Meara



我们的审校团队：

非常感谢我们的审校团队。Johannes de Jong组织并领导了这次成功的演出，他扮演了"丛书之父"的角色，而且使得整个工作变得畅通无阻。Pauline McNamara，整个审校工作的协调人，把各个部分联系一起，并首先指出部分例题过于庞大的弊病。整个审校团队在技术上的意见和细节上的关注对本书非常重要。Valentin Crettaz，Barney Marispini，Marcus Green，Ike Van Atta，David O'Meara，Joe Konior和Corey McGlone，在审校中不遗余力的检索，使得该书更趋完美。你们是好样的！还要感谢的是Corey和Pauline，令我们避免了过度呆板的说教，对JavaRanch的大叫也是我们工作生活中的一部分。

十分感谢Louise Barr，我们的网站开发者，她让我们在设计和使用XHTML及CSS时有了自信（尽管你还是会抱怨我们目前的设计）。

致谢*

更多的技术审校：

爱人尊敬的技术审校David Powers



不要让温柔的毛衣欺骗了你，他可是条硬汉（当然是技术上的）。

同样非常感谢我们尊敬的技术审校David Powers。我们对他可是又爱又恨。他督促我们拼命工作，虽然累，但是回报是巨大的。事实告诉我们，基于David的评论，我们对该书进行了意义重大的修改，使得该书的技术含量倍增，谢谢你，David。

致O'Reilly：

向我们的编辑Brett McLaughlin致意感谢，他牺牲了自己和家人共享天伦之乐的时间为本书扫清了前路的一切障碍。Brett在本书的编辑时期内非常努力（编辑Head First丛书可不是那么简单的工作）。谢谢Brett，如果没有你就没有这本书。



Brett McLaughlin

真诚地感谢整个O'Reilly团队：Greg Corrin，Glenn Bisignani，Tony Artuso和Kyle Hart做市场推广，我们很欣赏他们别出心裁的方式。感谢Ellie Volkhausen，她那灵感十足的封面设计使我们工作更有信心，还有Karen Montgomery，使书的封面更有生命力，同样感谢Colleen Gorman的校对工作（他以此为乐）。我们也不可能在没有Sue Willing和Claire Cloutier的情况下出版这么一本色彩斑斓的书。

致谢中不能缺少的是Mike Loukides，他将Head First这一概念引入丛书，还有Tim O'Reilly的不懈的支持。最后，感谢Mike Hendrickson，他令整个小组像家庭一样温暖，使我们坚定信念不断前进。

最后，少不了感谢Kathy Sierra和Bert Bates，我们的“共犯”和丛书的“大脑”。谢谢你们对我们的信任甚至多于你们的孩子。再次希望我们充分发挥了自己的能力。三天频繁的开会讨论像灯塔一样为我们写书确立了方向，我们迫不及待地希望下次的合作。哦，对了，你下次能不能打个电话给LTJ让他来西雅图旅行一趟？

Bert Bates

Kathy Sierra



努力工作

Kara

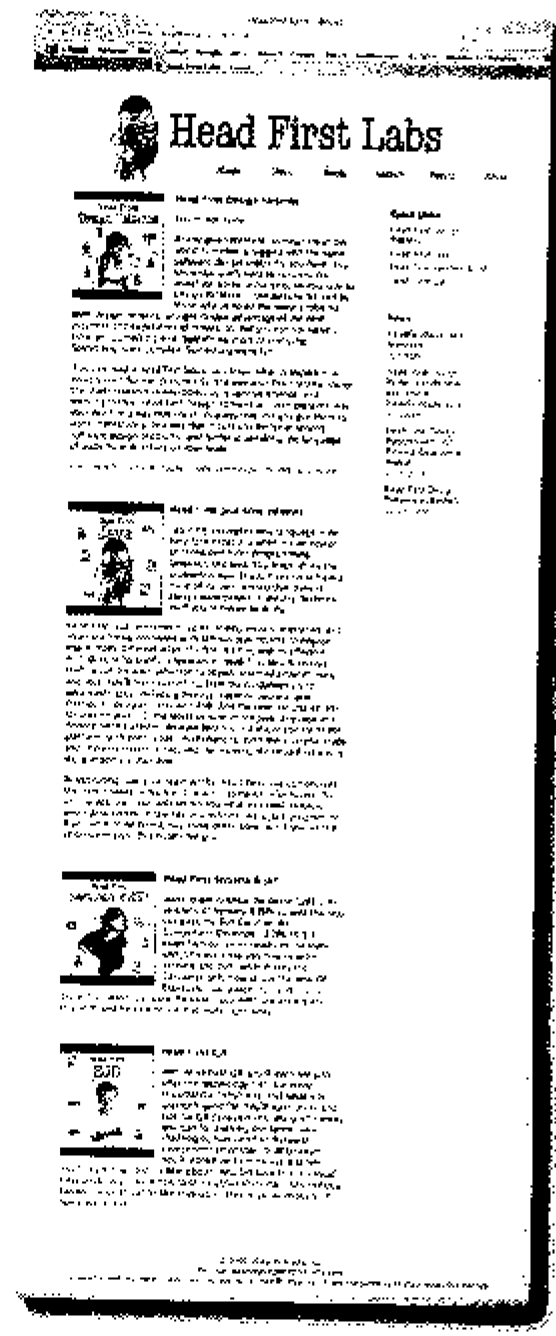
* 之所以要感谢这么多人，是因为我发现了这样一条定律，书中致谢部分里提到的每个人都至少会买一本书，可能还会买好几本书，给亲戚和周围的所有人都送上一本。如果你希望我们在下一本的致谢里提到你，而且你们家族的人很多的话，可以写信给我们。

Now that you've applied the Head First approach to HTML and CSS, why not apply it to the rest of your life?

快来加入我们吧！通过Head First虚拟实验室，你能够找到各种Head First资源，包括podcast、论坛等。

不仅这样，你不只是一位观众，你也能够通过参与讨论和头脑风暴获得很多乐趣。

现在你通过Head First的方式学习HTML和CSS，你的生活将变得很精采。在这里，你将得到：



我们为你准备了什么？

- 来自Head First世界的最新消息；
- 分享我们即将问世的图书与技术；
- 在极短的时间里解决技术上的难题；
- 了解Head First书籍幕后的故事；
- 与Head First作者与支持团队进行交流；
- 你还可以尝试成为Head First作者。

Why wait?
Our web servers
are standing by.
Visit now!

http://www.headfirstlabs.com

目录（概览）

	引子	xxv
1	Web语言：开始了解HTML	1
2	认识HTML中的"HT"：深入理解超文本	43
3	创建网页：构建模块	77
4	Web镇之旅：开始链接	125
5	认识媒体：给网页添加图像	165
6	严格的HTML：遵循标准，合乎规范	223
7	添加一个"X"到HTML：转到XHTML	265
8	添加一些样式：开始学习CSS	285
9	扩大你的词汇量：字体和颜色样式	341
10	与元素亲密接触：盒模式	385
11	高级网站构建：div和 span	429
12	布置元素：布局和排版	487
13	开始制作表格：表格和列表	549
14	交互活动：XHTML 表单	591
附录A：排名前十的主题（本书未涉及）：剩余部分		639

目录（真正的目录）

引子

让你的大脑来学HTML和CSS。 你想学些东西，但你的大脑却在帮倒忙，不让你记住这些东西。你的大脑在想："还是把空间留给更重要的事情吧，比方说要躲开哪些野兽，还有，光着身子滑雪不太好吧。"那么你该如何骗过大脑，让它认为要是不知道HTML和CSS你就活不下去了？

谁适合读这本书？	xxvi
我们知道你的大脑在想什么	xxvii
元认知	xxix
让你的大脑就范	xxxi
技术审校	xxxiv
致谢	xxxv

1 开始了解HTML

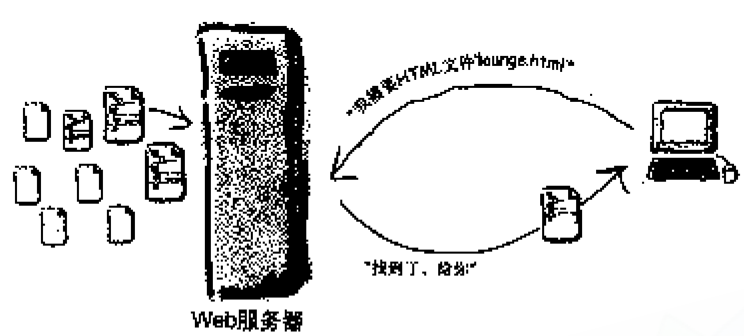
Web语言

你和Web之间的唯一障碍就是学习使用其中的语言：

HyperText Markup Language（超文本标记语言），简写为HTML。因此，你要准备好一些语言课程。学完这章后，你不但懂得基本的HTML组成元素，而且还能编写具有简单样式的HTML语句。当学完本书后，你就会像说母语一样自如地运用HTML。



Web击败了video明星 2
Web服务器能干些什么？ 3
让我们（用HTML）来写些什么吧…… 4
浏览器创建了什么…… 5
你在Starbuzz咖啡馆走了好运 9
建立Starbuzz网页 11
创建一个HTML文件（使用MAC系统） 12
创建一个HTML文件（使用Windows系统） 14
现在，让我们回到Starbuzz的工作上来…… 17
用浏览器打开你的网页 19
调试你的网页…… 20
解剖标记…… 25
认识样式元素 29
给Starbuzz添加样式…… 30
谁做了什么？ 32
围炉夜话 34
要点 36
习题解答 38

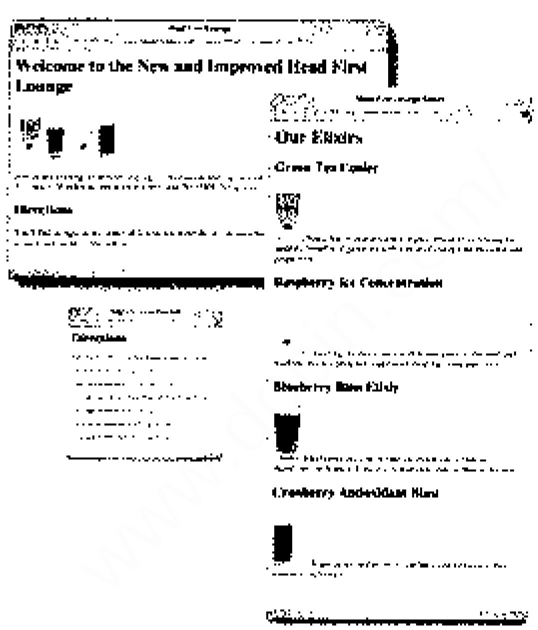


2

深入理解超文本

认识HTML中的"HT"

有人提到"hypertext"吗？那是什么？哦，它是整个Web的基础。在第1章我们开始学习HTML，并知道了它是一种优秀的描述网页结构的标记语言（即HTML中的'ML'）。现在我们准备学习HTML中的'HT'［即超文本（hypertext）的缩写］，它将引领我们把一个网页链接到其他网页。现在我们学习一个新的元素——<a>元素，并了解"相对"是个多么有用的东西。现在，系紧你的安全带，马上进入超文本的学习。



崭新且改进过的Head First休闲室 44

创建新的休闲室 46

我们做了些什么？ 48

浏览器要做什么？ 49

了解属性 51

技术难点 58

计划好你的路径…… 60

修复损坏的图像…… 66

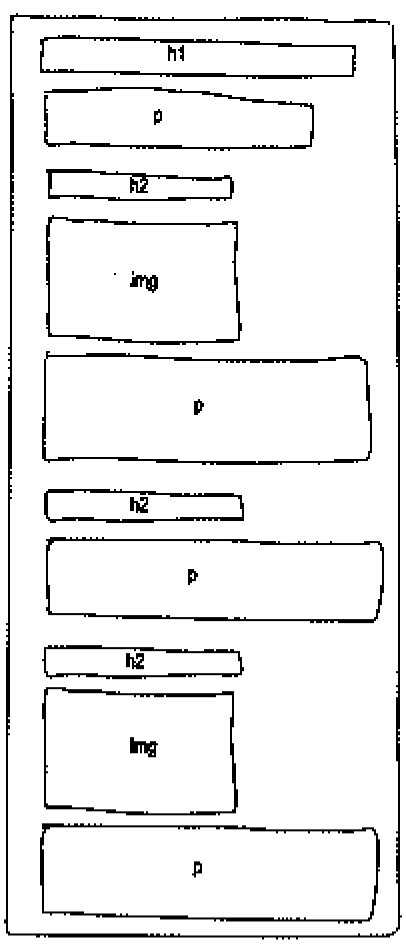
习题解答 73



3 构建模块

创建网页

听说我能从这本书中学到如何创建网页？你已经学会很多了：标记、元素、链接、路径……但是如果不能学以致用，那一切都是徒劳。在这一章，你要把网站由概念变成蓝图，浇注根基，构建它，再添加些格调。你的任务是做好准备，因为我们将引入些新工具，帮你学习更深入的知识，使你工作起来得心应手。



以每小时12公里的速度启动网站之旅 79

粗略的设计草图 80

从草图到略图 81

从略图到网页 82

测试Tony的网页 84

认识<q>元素 86

很长很长的引用 90

添加<blockquote> 91

<q>和<blockquote>奇案背后的真相 94

用<p>元素来做一个列表…… 103

简单的两步创建HTML列表 104

把一个元素放进另一个元素，叫做“嵌套” 109

用图示来了解嵌套的关系 110

使用嵌套来确保你的标记匹配 111

内联元素还是块元素？ 113

习题解答 119

4 开始链接

Web镇之旅

网页是互联网给我们的最好的礼物。迄今为止，你已经学会了创建只能在自己电脑里生存的HTML页，还只能链接到你电脑中的其他页。我们将要来个翻天覆地的改变。在这一章我们鼓励你把这些网页搬到网上去，你的朋友、fans和顾客都可以看到。我们同样会通过破译代码h，t，t，p，：，/，/，w，w，w等揭示链接到其他网页的秘密。那么，收拾好你的行李，下一站就是Web镇。

发布Starbuzz（或者你自己的）主页到Web上 126

寻找主机代理商 127

如何获得一个域名？ 128

搬迁 130

把你的文件复制到根目录 131

尽量在两页中介绍你可能用到的FTP 132

返回休闲室主页…… 135

关键的URL 136

什么是HTTP协议？ 137

什么是绝对路径？ 138

默认网页如何工作 141

怎样链接到其他网页？ 144

链接到Caffeine Buzz 145

网页修饰并完成 149

链接到一个页面 151

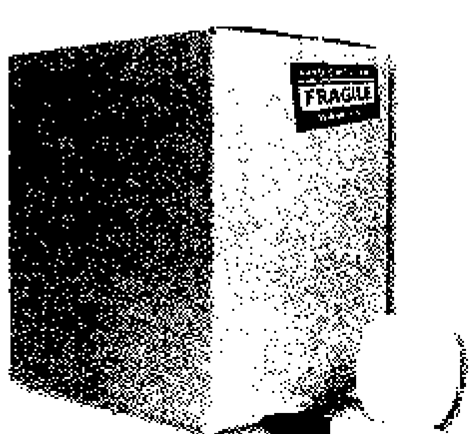
使用<a>元素创建目的地 152

如何链接到目标锚 153

链接到一个新窗口 157

使用target（对象）打开一个新窗口 158

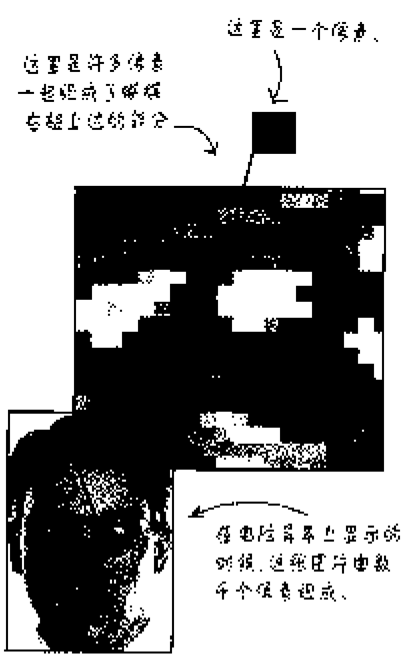
习题解答 162



5 给你的网页添加图像

认识媒体

微笑着说“茄子”。事实上应该是微笑着说“gif”、“jpg”、“png”——当为网页添加图片时，你将会用到这些。在这一章，你将学会在网页中添加第一种媒体类型：图像。你想在网上获得你想要的数码相片？没问题。想为你的网页添加一个logo？没问题。但这之前，我们要再次介绍<img>元素。真是抱歉，并非我们粗心大意，而是我们从来都没“正式地介绍”。为了补偿这一点，我们会用这整整一章来介绍它。学完本章后，你会知道如何使用<img>元素和它的属性等所有细节。你同样会看到这个小元素怎么让浏览器做大量额外的工作来接收并显示图像。



浏览器如何处理图像	166
图像是如何工作的	169
现在正式介绍：认识<img>元素	173
总是提供可选项	176
创建最后的fan网站：myPod	178
哇！图像太大了	181
修改myPod HTML	191
使用缩略图重新修改网站	195
让缩略图变成链接	199
怎么创建图像的链接呢？用images?	201
我们应该用哪种格式？	206
透明还是不透明？这是个问题……	207
等等，什么是网页的背景颜色？	209
用蒙版做logo	210
添加logo到myPod网页	211
习题解答	216

6 标准，规范，还有那些夸夸其谈

严格的HTML

关于HTML，还有什么知识需要了解呢？你已经熟练地掌握了一部分HTML。接下来我们开始学习CSS。通过它，我们可以让所有乏味的标记变得生动有趣。不过，首先要严格规范我们编写HTML的方法，以确保你的HTML结构严谨。别误会，你一直在使用一流的HTML，不过你还可以做点别的让浏览器如实地显示你的网页并确保标记没有错误。这对你有什么好处呢？网页可以在浏览器上得到更统一的显示（甚至在移动设备和为视觉障碍用户提供的屏幕读取器上能更好地显示），网页导入更快，并能保证网页更好地与CSS结合。准备好，在这一章我们将从制作网页的菜鸟转变为高手。

办公室隔间对话 224

HTML简史 226

我不会让你的网页引导浏览器进入转换显示模式 229

添加文件类型定义 231

接触W3C 校验器 234

校验Head First休闲室 235

Houston，我们遇到问题了…… 236

添加<meta>标记说明内容的类型 240

让校验器更容易识别<meta>内容标记…… 241

第三次尝试的魅力 242

把DOCTYPE改为严格版本 246

是否被认可了? 247

解决嵌套问题 249

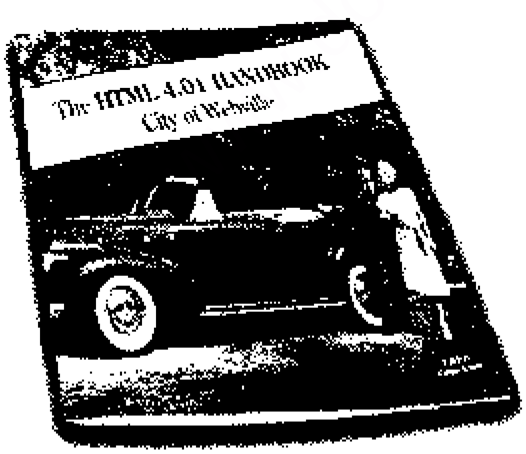
再次测试是否严格了…… 250

严格的HTML 4.01，紧扣手册 252

围炉夜话 256

HTML考古 259

习题解答 263



7 转到XHTML

添加一个“X”到HTML

我们一直对你隐瞒着一个秘密。你以为自己买了一本HTML书，但其实这是一本化了妆的XHTML书。事实上，一直以来我们教你的大多是XHTML。你一定很好奇，XHTML究竟是什么？好了，来认识HTML的扩展版本——即所谓的XHTML——HTML的下一版本。它更简单、通俗，并且在一系列的浏览器上更具兼容性。在这简短的一章里，我们将通过三个步骤，由HTML过渡到XHTML。好了，现在翻开书本，我们要开始学习了……（之后我们将学习CSS）。

什么是XML？	267
如何将HTML改为XHTML？	268
为什么要使用XHTML？	270
规范XHTML的校验表	272
严格的HTML转化为XHTML1.0	274
旧版的HTML4.01	275
新的改进了的HTML4.01	275
校验：不只适用于HTML	277
围炉夜话	280
HTML还是XHTML？决定权在于你……	282
习题解答	284



8 开始学习CSS

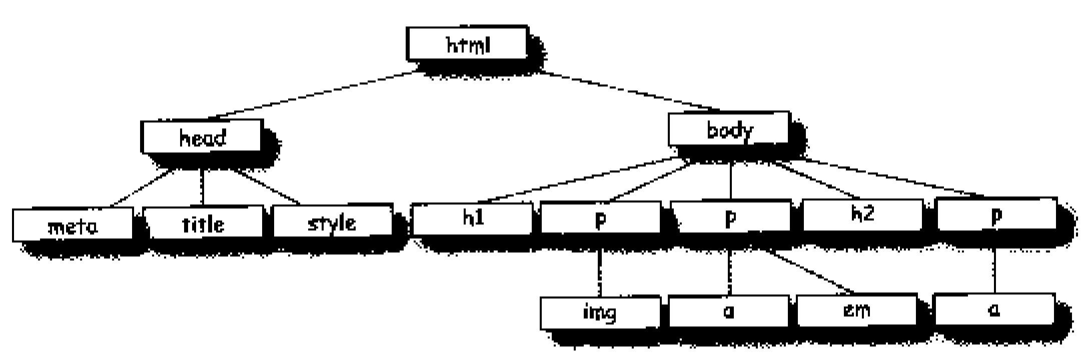
添加一些样式

听说这本书将介绍CSS。迄今为止，你一直专注于学习用XHTML创建网页结构。但是如你所见，浏览器对网页的样式有很高的要求。当然，你可以借助一些工具，但那是不必要的。使用CSS，你可以完全控制网页的外观，甚至不用更改XHTML。它真的可以这么容易就能实现吗？好了，你将要再学一门新的语言，毕竟，在Web镇上可以同时使用两种语言。读完本章对CSS语言学习的介绍，相信你会对网页设计有更全面的了解。

你不再处于Kansas了……	286
在Web镇“贸易区”之见闻	288
使用CSS设计XHTML	289
给欢迎词加下划线	295
指定只针对<h1>的规则	296
对elixir 和directions 页面进行同Lounge一样的设计	303
把“lounch.html”链接到外部样式表	305
了解继承……	311
如果我们把字体属性上移会有什么结果?	312
继承的覆盖	314
创建类的选择符	318
进一步了解类	320
样式应用最快捷最小巧的手册	322
谁继承了属性?	326
确保休闲室的CSS合法	329
习题解答	333

五分钟之谜



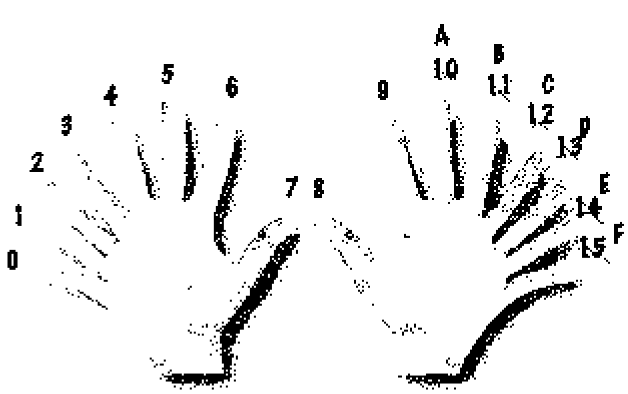


9 字体和颜色样式

扩大你的词汇量

愉快的CSS语言课程就要开始了。你已经了解了CSS的基础，知道如何创建CSS规则来选择和定义一个元素的样式。现在该扩大你的词汇量了，这意味着你要学习一些新的属性及它们的功能。在这一章中我们将用那些能影响文本显示的、最普通的属性来完成工作。为此，你需要学习一些关于字体和颜色的知识。你将会发现你不必始终只用大家都用的字体，或浏览器默认的段落和标题中粗笨的字体大小和样式。你会发现更多新奇的颜色。

从一定的高度看文本和字体	342
到底什么是字体系列呢？	344
用CSS定义字体系列	347
重新看一下Tony的日志	348
每个人都会有不同的字体，我该怎么处理？	351
那么，应该用哪种方法定义我的字体大小呢？	354
我们来改变一下Tony网页中的字体大小	356
改变字体粗细	359
给字体添加样式	361
把Tony网页中的引用变成斜体	362
Web颜色如何工作？	364
如何指定网页颜色？方法有……	367
快速了解十六进制代码	370
如何确定Web颜色	372
回到Tony的页面……	375
你想知道的关于text-decoration的所有内容	377
去掉下划线……	378
习题解答	381

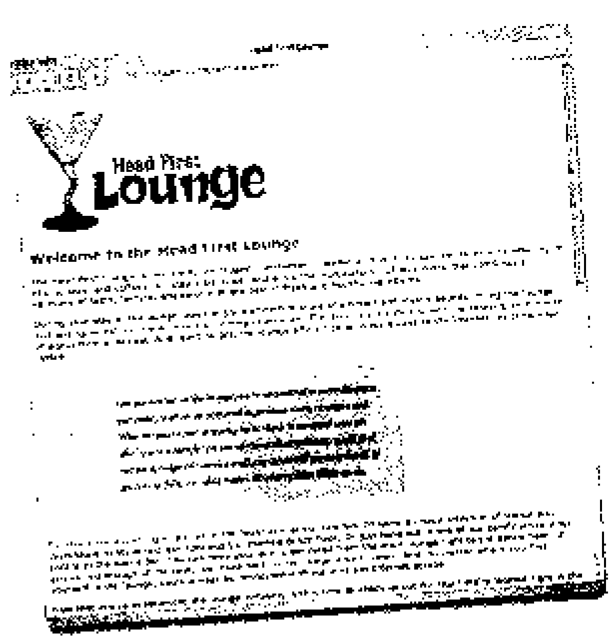


10 盒模式

与元素紧密接触

要更好地构建网页你必须了解你的建筑材料。在这章中我们将仔细观察我们的建筑材料：XHTML元素。我们将仔细研究块和行元素，看看它们由什么组成。你将看到如何用CSS构建元素来控制各个方面。但我们不会止于此——你还会知道如何给元素指定唯一的标识符。如果这还不够，你还将学习什么时候及为什么要用多样式表。好的，翻开本页，让我们与元素来个亲密接触吧。

改进休闲室	386
创建新休闲室	388
仔细观察新行间距	390
准备大翻新	391
进一步解析盒模式（box model）……	392
你能对盒子做什么？	394
创建保证样式	399
设置保证段的补白、边框和边界	401
添加 些补白	401
添加一些边界	402
插入背景图像	404
搞定背景图像	407
如何只在左侧添加补白？	408
如何只在右侧增加边界？	409
边框快速指南	410
确定边框并完成	412
访问HTML的类	414
id属性	416
在休闲室中使用id	418
重新混合样式表	420
使用多样式表	421
习题解答	426

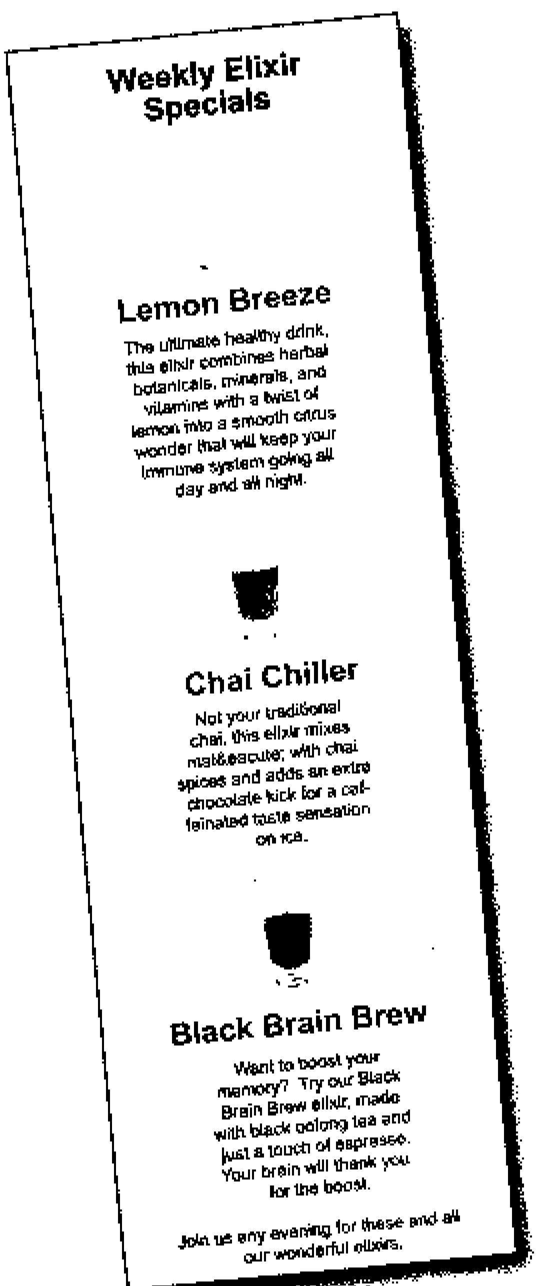


www.docin.com

11 div和span

高级网站构建

该来点高级建筑了。这一章我们将引出两个新的XHTML元素，叫做<div>和<span>。这些不是“微不足道”的小东西，而是到处都要用到的钢铁横梁。你将用<div>和<span>构建一些严格的支持结构，一旦把这些结构放到合适的位置，就能用全新而强大的方式样式化它们了。现在，我们还不能帮你这个忙，因为我们发现你的CSS工具箱已经开始填满了，所以首先要告诉你一些捷径，以便更加轻松地指定这些属性。此外，这一章有一些特别的客人——伪类，你可以用它们创建一些非常有趣的选择符（如果你觉得“伪类”可以作为你下一个乐队的名字，那么太晚了，我们抢先了一步）。



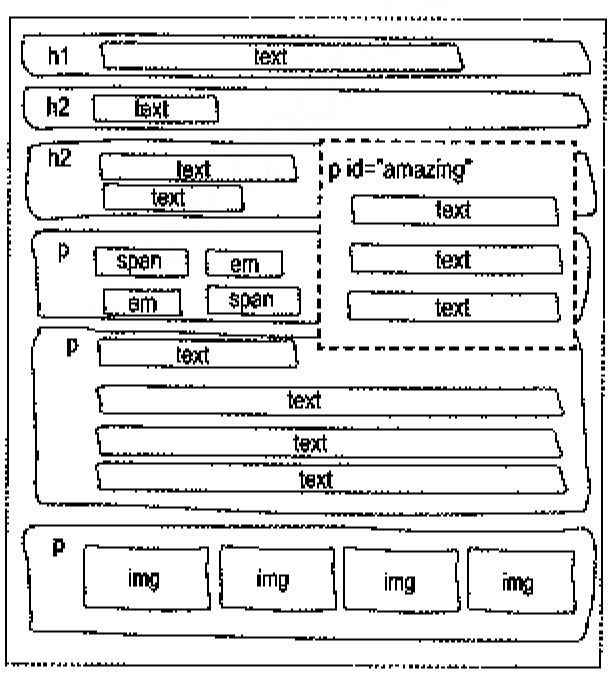
仔细观察饮料XHTML	431
研究一下如何把一个页面分割成几个逻辑部分	433
添加边框	440
测试边框	440
给饮料部分添加些真正的样式	441
游戏计划	442
调整饮料部分宽度	442
给饮料添加基本样式	447
我们需要一种选择子孙的方法	453
改变饮料标题的颜色	455
确定行间距	456
走点儿捷径	458
用简单的三步添加<span>	464
<a>元素和它的几个特征	468
如何根据状态样式化元素?	469
应用伪类	471
该介绍“层叠（cascade）”了吧?	473
层叠	475
欢迎来到“我的具体度是什么”游戏环节	476
融会贯通	477
习题解答	483

12 布局和排版

布置元素



再来教你们一些关于XHTML元素的新方法。我们不再只是让XHTML元素呆坐在那儿……是时候让它们起来帮我们创建一些有真正布局的页面了。怎么样？好的，你已经很熟悉<div>和<span>这两个划分结构的元素了，并且完全了解盒模型是如何工作的，对吗？那么，现在就该用所学知识来做一些真正的设计了。现在不仅介绍关于背景和字体颜色的内容，还要介绍如何用多栏布局进行专业设计。这一章中要用到所有学过的东西。



做思考题了吗？ 488

Luke，使用流 489

对于内联元素呢？ 491

这些如何一起工作 492

如何漂移元素 495

在休闲室后台 497

新的Starbuzz 499

把sidebar移到标题下 504

设置sidebar的宽度并漂移它 504

搞定分栏问题 507

设置main部分的边界 508

回来处理重叠问题 511

右紧左松 514

流动和冻结设计 517

绝对布置如何工作？ 520

改变Starbuzz CSS 523

固定页脚的折衷办法 527

放置奖杯 529

固定布置如何工作？ 535

使用负的left属性值 537

了解相对布置 539

做成三栏甚至更多栏…… 541

习题解答 544

13 表格和更多列表

开始制作表格

如果像表格那样行走和交谈……生活中，我们常常要处理一些令人头痛的表格式数据。不管是创建一个描述你公司去年的存货清单的网页，还是制作一个布娃娃收藏的编目表（别担心，我们不会说出去），你都需要用XHTML来实现；但如何实现呢？在这一章中我们将解读表格的秘密让你将个人数据正确地存入XHTML表格。此外，如果你现在就行动，作为特殊的奖励，我们将抛出指南引导你设计XHTML列表。别犹豫，行动吧！

如何用XHTML创建表格？ 551

用XHTML创建一个表格 552

浏览器生成了什么？ 553

表格剖析…… 554

添加标题和摘要 557

样式化表格之前，先把表格放到Tony的网页上 559

压缩边框 564

对于颜色呢？ 566

Tony有一个有趣的发现…… 567

从另一个角度看Tony的表格 568

将单元格扩展为多行 569

新型的改进了的表格 571

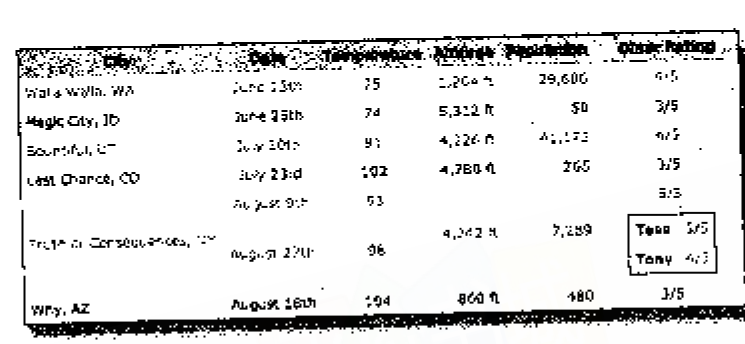
天堂的烦恼？ 572

为嵌套的表格表头覆盖CSS 576

Tony网页的最后润饰 577

习题解答 588

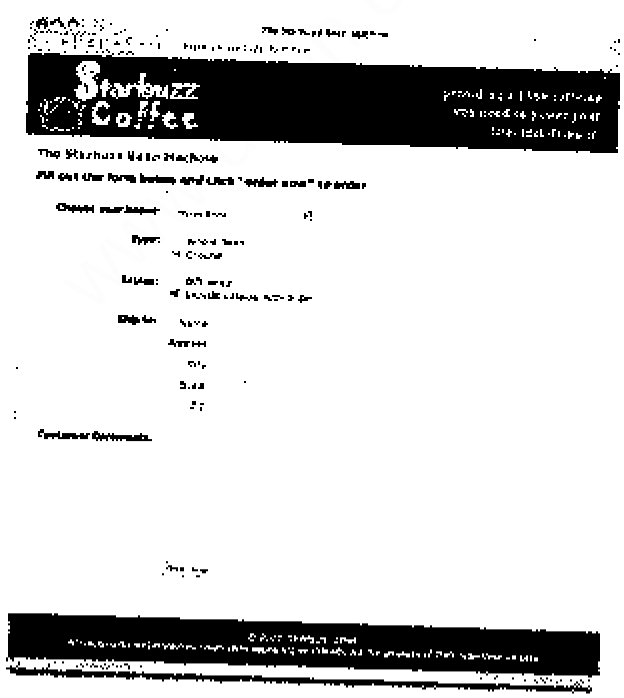




14 Xhtml表单

交互活动

迄今为止，Web之间的通信都只是一种方式：从网页到访问者。如果访问者能够反馈，肯定非常棒，是不是？因此我们引进了XHTML表单：你一旦在网页中添加表单（同时需要Web服务器的帮助），你的网页就能够收集客户的反馈，提取订单，或者转到下一个在线游戏，或者统计热门或冷门的竞赛投票。在这一章中，你会碰到一个小队的XHTML元素，这些元素一起创建了Web表单。你还可以了解一些后台服务器处理表单的知识，此外，我们还将讨论表单的更新（一个有争论的话题，读下去会找到原因）。



表单如何工作	592
表单在浏览器中如何工作	593
用XHTML写些什么	594
浏览器生成了什么	595
表单元素如何工作	596
准备创建Bean Machine表单	604
添加表单元素	605
表单元素的名如何工作	606
将<input>元素加到XHTML中……	608
在表单中添加更多的<input>元素	609
添加<select> 元素	610
给顾客一个选择，是whole（全）咖啡豆呢还是ground（研磨的）咖啡豆	612
显印单选框按钮	613
完善表单	614
添加复选框和文本区	615
运行观察GET	621
用不用表格？这是个问题……	626
把表单元素放入表格	627
用CSS样式化表单和表格	630
习题解答	635

15 附录A：剩余部分

排名前十的主题（本书还未涉及）

我们介绍了许多基础知识，而且你也差不多读完了此书。我们会想念你的，不过在让你走之前，如果就这样在你毫无准备的情况下将你推向社会，我们觉得不太合适。我们不可能把所有你需要了解的知识都包含在这一小节中。事实上，我们最初是这么做的，把字体尺寸减少到.00004，这样的话这一小节就包括了所有你需要知道的关于XHTML和CSS的知识（其他章节中没有讲到的）。这样确实能够把所有内容装进去，不过恐怕也没有人会读了。所以，我们扔掉了大部分内容，只保留了最好的排名前十的主题。



更多选择符 640

框架 642

多媒体和Flash 643

创建网页的工具 644

客户端脚本 645

服务器端脚本 646

转到搜索引擎 647

关于打印的样式表 648

适合移动设备的网页 649

blog（博客） 650

i 索引 651

1 开始了解HTML

Web语言



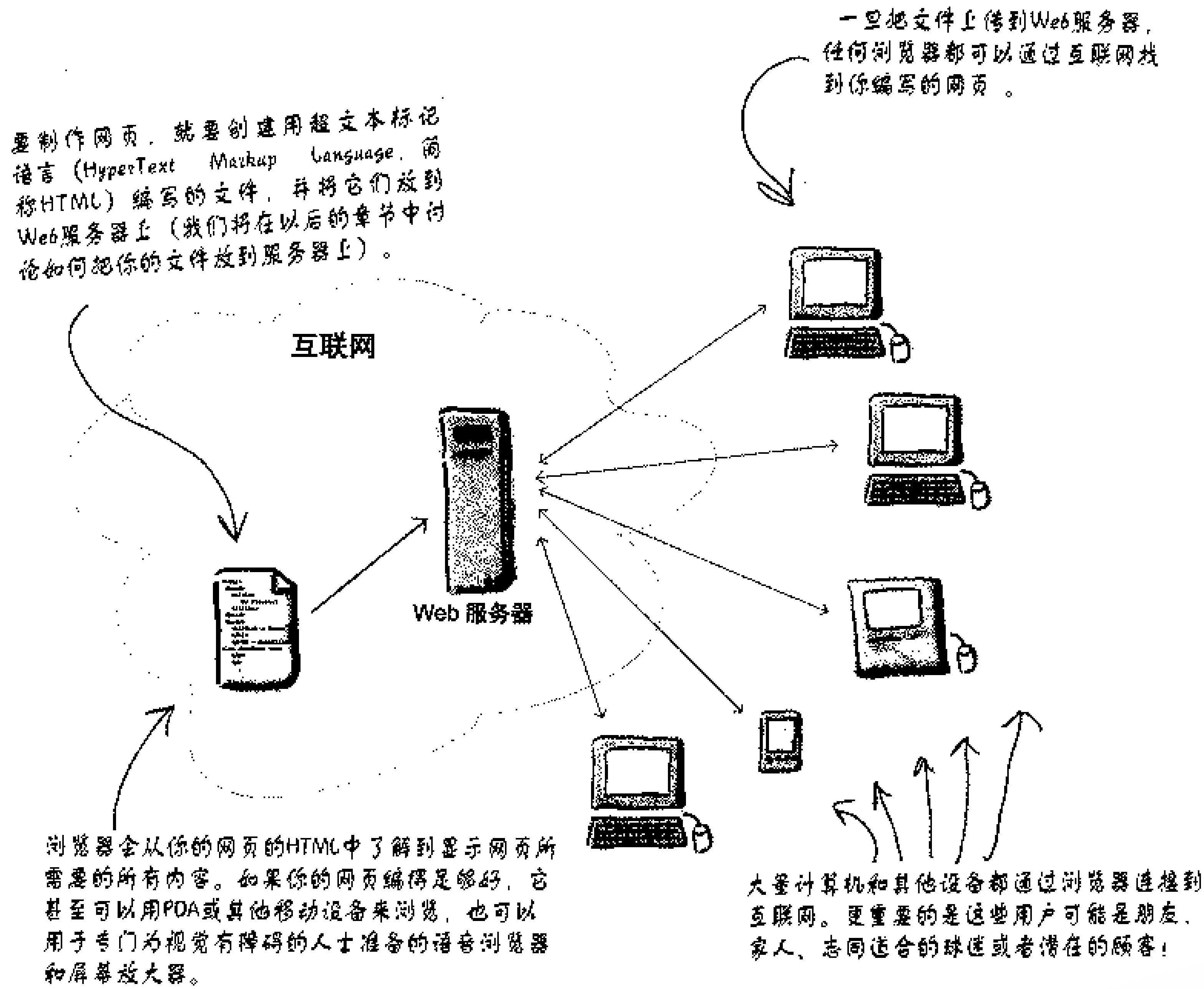
你和Web之间的唯一障碍就是学习使用其中的语言：

HyperText Markup Language（超文本标记语言），简写为HTML。因此，你要准备好学习一些语言课程。学完这一章后，你不但会懂得基本的HTML组成元素，而且还能编写具有简单样式的HTML语句。当学完本书后，你就能像说母语一样自如地运用HTML。

电视击败了电台明星

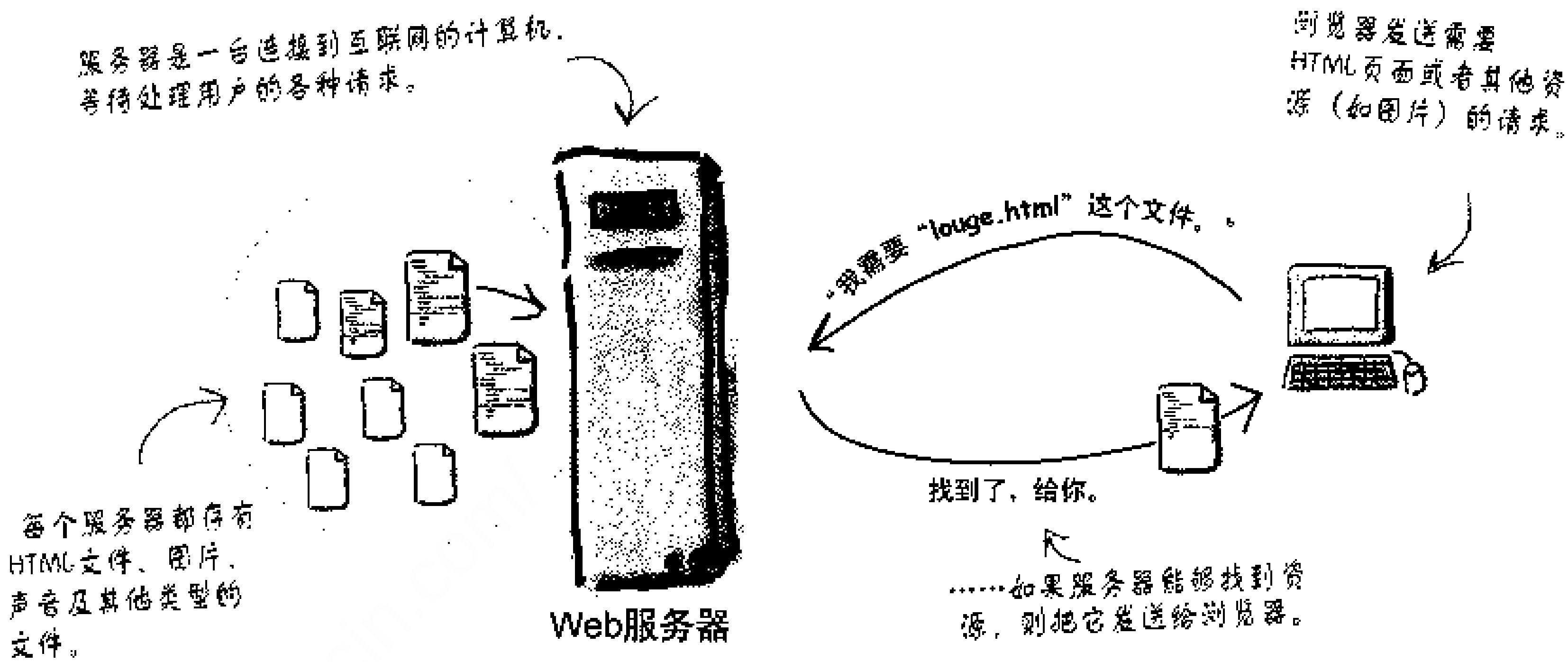
想表达自己的观点？兜售物品？还是仅仅秀一下自己的创作？交给Web吧——毋庸置疑，Web已经成为全球化的交流方式。而且，你也能参与其中。

然而，如果你想有效地利用Web，就得了解一些关于HTML的事情，以及Web的工作原理。先让我们从宏观的角度看看：



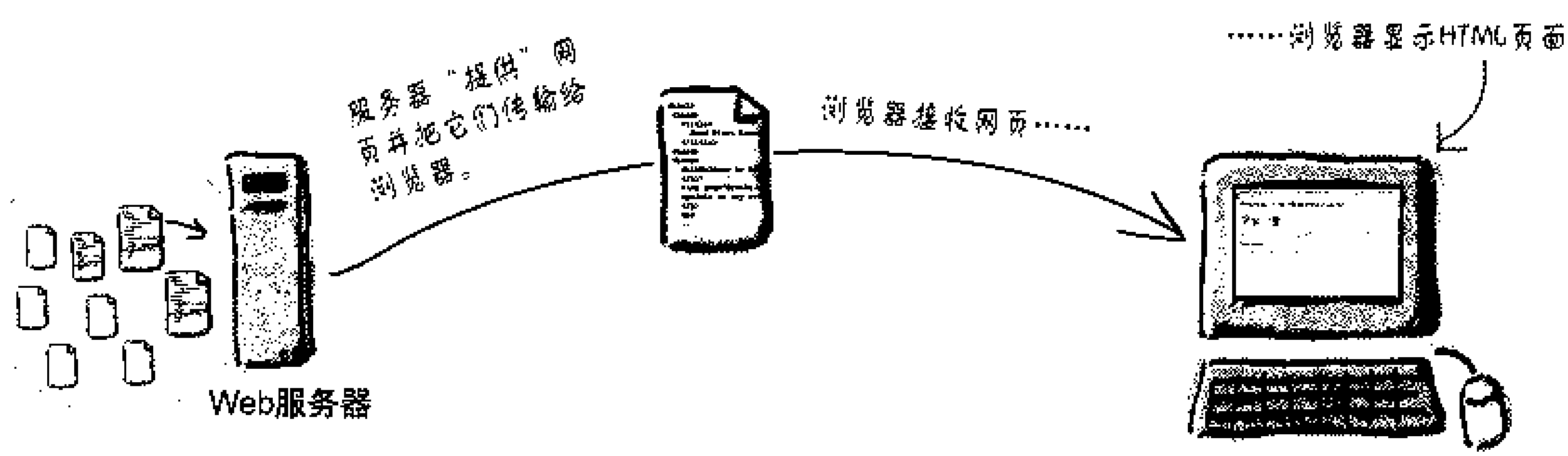
Web服务器能干些什么？

Web服务器在互联网上日以继夜地工作，不知疲倦地等待处理浏览器的各种请求。都是些什么请求呢？有浏览网页、图片，播放声音，甚至观看电影等。当服务器接到针对这些资源的请求时，它将找到资源，然后，将其发送到相应的浏览器上。



Web浏览器能干些什么？

你已经大致了解了浏览器的工作原理：你到网上冲浪并点击某个链接来浏览网页。你的点击导致浏览器向Web服务器请求一个HTML网页，然后，浏览器接收该网页，并在浏览器窗口上显示它。



然而，浏览器是怎么知道如何显示一个页面的呢？这就是HTML存在的意义。HTML告诉浏览器页面的结构和内容。让我们看看它是如何工作的……

让我们（用HTML）写些什么吧……

好了，你知道了HTML是浏览器显示网页的关键。那么HTML到底是什么样子呢？还有，它是干什么的呢？

让我们来看个简单的HTML例子，假设你想为Head First休闲室创建一个宣传网页。Head First休闲室是一家具有美妙音乐、新鲜饮料及无线接入的休闲场所。下面是你编写的HTML语句。

```
<html>
  <head>
    <title>Head First Lounge</title>      (A)
  </head>
  <body>
    <h1>Welcome to the Head First Lounge</h1>      (B)
    <img src="drinks.gif">      (C)
    <p>
(D)   Join us any evening for refreshing elixirs,
      conversation and maybe a game or
      two of <em>Dance Dance Revolution</em>.      (E)
      Wireless access is always provided;
      BYOWS (Bring your own web server).
    </p>
    <h2>Directions</h2>      (F)
    <p>
(G)   You'll find us right in the center of
      downtown Webville. Come join us!
    </p>
  </body>
</html>
```



放轻松

我们还不指望你马上理解HTML。

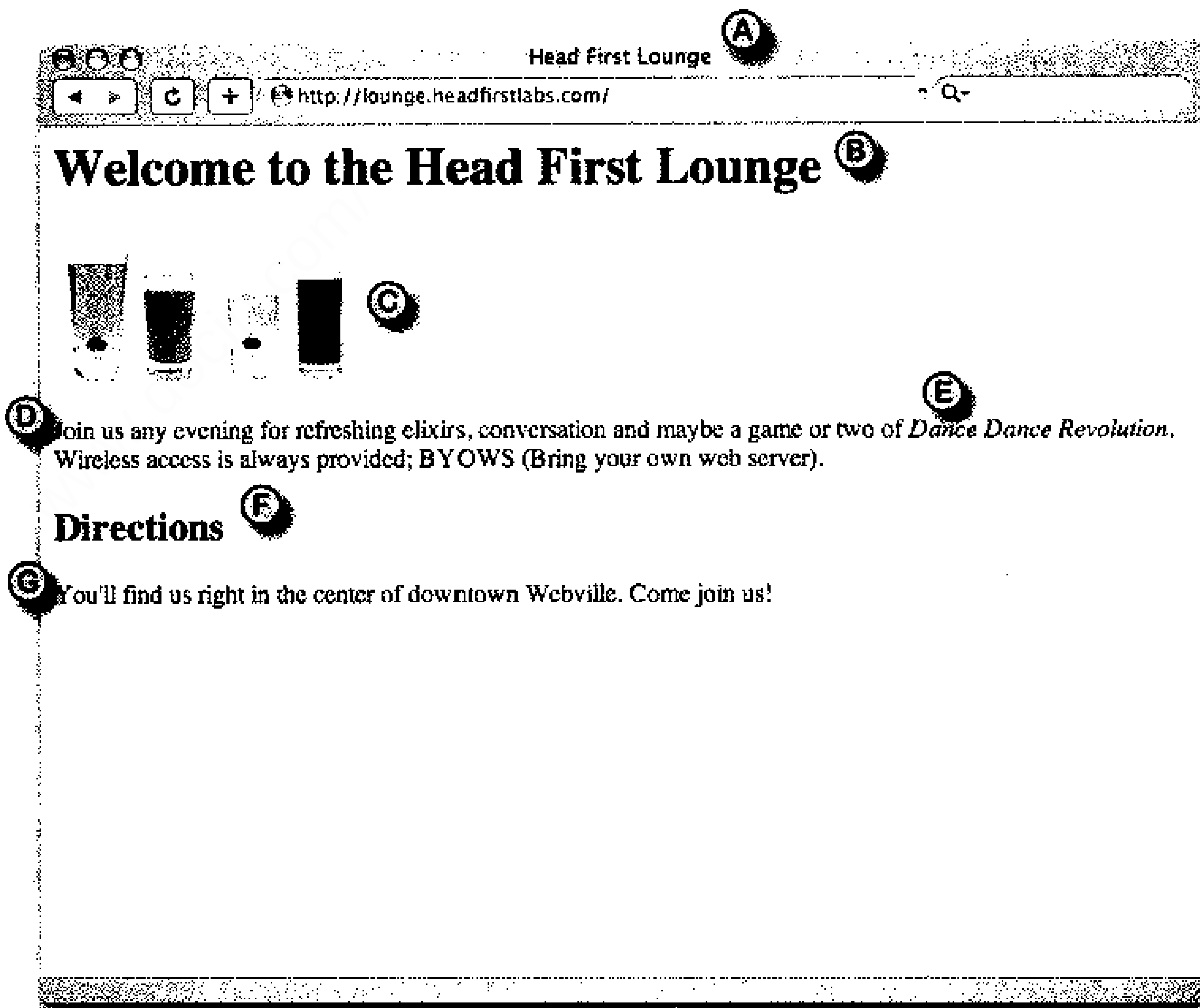
我们只是让你熟悉一下HTML的格式，之后我们将循序渐进地逐步展开。现在，学习HTML，并注意观察它在浏览器中的显示情形（参见下一页）。特别要关注附有字母注释的部分，以及它们在浏览器中显示的结果和位置。

浏览器创建了什么……

当浏览器阅读HTML时，它会解析包围文本的所有标记。标记是用尖括号括起来的字母或者单词，例如**<head>**、**<p>**、**<h1>**等。标记告诉浏览器文本的结构和意义。因此，这不是只给浏览器一堆文本。通过HTML，你能使用标记来告诉浏览器哪些文本是标题，哪些文本是段落，哪些需要强调，甚至哪里需要放置图片。

让我们来看看浏览器是怎么解析Head First 休闲室中的标记的。

注意HTML中的每个标记对应浏览器中显示的内容。



there are no Dumb Questions

问： **可以说HTML就是一大堆包围文本的标记吗？**

答： 对初学者来说是这样的。记住HTML是超文本标记语言的缩写。HTML提供用标记标明文本的方式，这种方式告诉浏览器文本的结构。但是HTML中还有超文本方面的内容，这些我们稍后会谈到。

问： **浏览器是如何显示HTML的？**

答： HTML告诉浏览器文档的结构：哪里是标题，哪里是段落，哪里需要强调等。在得到以上信息后，浏览器会按照默认的规则来显示每个元素。

然而，你不必勉强接受默认的设置。你能够通过CSS来添加自己的样式和格式规则，从而决定字体、颜色、大小及其他个性化的特征。我们会在后边的章节介绍CSS。

问： **Head First休闲室中的HTML里有许多缩进和空格，但在浏览器中却看不到它们，这是为什么？**

答： 这个问题问得好。浏览器显示HTML文件时会忽略制表符、回车、还有大多数的空格。它们根据你的标记来确定哪里分行或者分段。

既然浏览器要忽略这些，我们为什么还要插入自己的格式呢？目的是为了帮助我们在编辑HTML时更易读懂该文档。随着文件越来越复杂，你将发现那些空格、回车还有制表符对提高HTML文件的可读性是非常有帮助的。

问： **为什么有两级标题，<h1>和一个子标题<h2>？**

答： 实际上一共有6个，从<h1>到<h6>，浏览器相应的字体也由大到小。除非你在写一个庞大复杂的文档，否则一般用不到<h3>以后的标题。

问： **为什么我要用<html>这个标记？这就是一个HTML文件，难道还不够明显吗？**

答： <html>标记会通知浏览器这是个HTML文件。有些浏览器允许省略该行，有些却不能，而我们是向"工业化加强版HTML"（书中稍后提到）看齐的，到时你会明白写上该行是非常重要的。

问： **是什么使得一个文件成为HTML文件？**

答： 基本来说，HTML文件是个简单的文本文件，但它和文字处理文件不同，并不内嵌特殊的格式。按照惯例我们用后缀".html"或".htm"（在某些只支持三个字母做文件扩展名的系统中）来告知操作系统该文件的类型。如你所见，文件的内容才是至关重要的。

问： **标记看起来有些落后。所见即所得的应用程序大约在20世纪70年代就出台了，为什么网站不采用像Microsoft Word或其他类似的应用程序一样的格式呢？**

答： Web是建立在没有任何特殊格式字符的文本文件上的，这使得各种浏览器在世界上的任意地方都能接收网页并理解其内容。你将发现在Web上HTML在许多方面都比其他专有文件格式具有优势。

问： **有没有把自己的注释放到HTML中的方法？**

答： 有，如果用标记<!--和-->把注释括起来，浏览器就会把它们全部忽略掉。如果你想加入"Here's the beginning of the lounge content"这段注释，可以这么做：

```
<!--Here's the beginning of the lounge content-->
```

你可以把注释写成多行，任何写在"<!--"和"-->"之间的内容都会被浏览器忽略。

Sharpen your pencil

你比你想象的更接近HTML……

以下还是那段Head First休闲室的HTML。注意那些标记，看看你能不能猜出它们分别告诉浏览器干些什么。把你的答案写在右边的空行上，我们已为你填上了前两个。

```
<html>                                                  告诉浏览器HTML是从这里开始的……
  <head>                                                页面"头"开始（稍后再讲）。
    <title>Head First Lounge</title>                    ____________________
  </head>                                               ____________________
  <body>                                                ____________________
    <h1>Welcome to the Head First Lounge</h1>           ____________________
    <img src="drinks.gif">                              ____________________
    <p>                                                 ____________________
      Join us any evening for refreshing elixirs,
      conversation and maybe a game or
      two of <em>Dance Dance Revolution</em>.           ____________________
      Wireless access is always provided;
      BYOWS (Bring your own web server).
    </p>                                                ____________________
    <h2>Directions</h2>                                 ____________________
    <p>                                                 ____________________
      You'll find us right in the center of
      downtown Webville. Come join us!
    </p>                                                ____________________
  </body>                                               ____________________
</html>                                                 ____________________
```

Sharpen your pencil
答案

```
<html>                                                  告诉浏览器HTML从这里开始。
  <head>                                                页面“头”开始。
    <title>Head First Lounge</title>                    给页面一个标题。
  </head>                                               头部结束。
  <body>                                                主体开始。
    <h1>Welcome to the Head First Lounge</h1>           告诉浏览器这句是标题。
    <img src="drinks.gif">                              在这里放置图片drinks.gif。
    <p>                                                 开始一个段落。
       Join us any evening for refreshing elixirs,
       conversation and maybe a game or
       two of <em>Dance Dance Revolution</em>.          强调Dance Dance Revolution。
       Wireless access is always provided;
       BYOWS (Bring your own web server).
    </p>                                                结束段落。
    <h2>Directions</h2>                                 告诉浏览器“Directions”是个子标题。
    <p>                                                 开始另一个段落。
       You'll find us right in the center of
       downtown Webville. Come join us!
    </p>                                                结束段落。
  </body>                                               结束主体。
</html>                                                 告诉浏览器HTML在这里结束。
```

你在Starbuzz咖啡馆走了好运



Starbuzz是家新兴的咖啡馆，因其不断开张的连锁店而日益出名。你可能在街角看到一家，而在马路的对面又是一家。

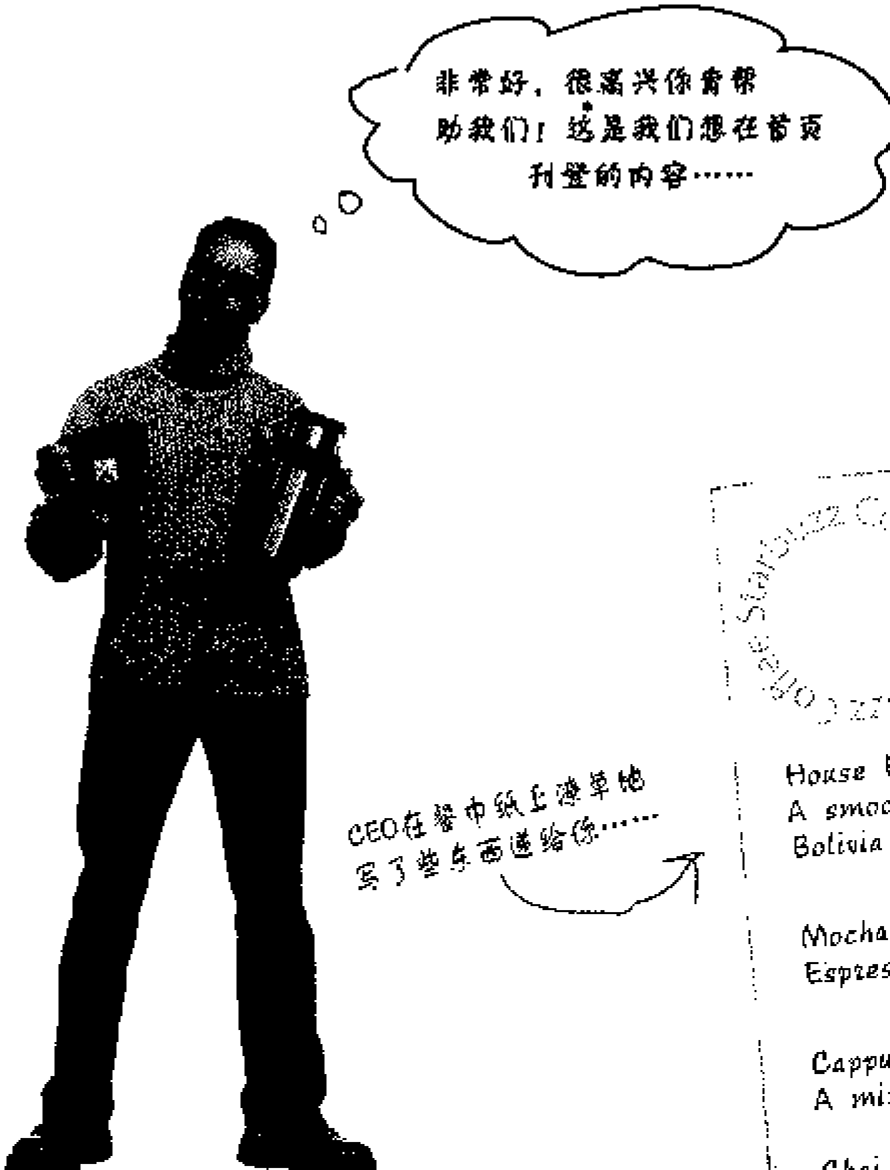
事实上，尽管咖啡馆发展迅速，但他们却还没有建立起自己的网站……这给了你一个机会。有一次，你到Starbuzz咖啡馆去买ChaiTea（印度香辣奶茶）时恰好碰到Starbuzz的CEO……

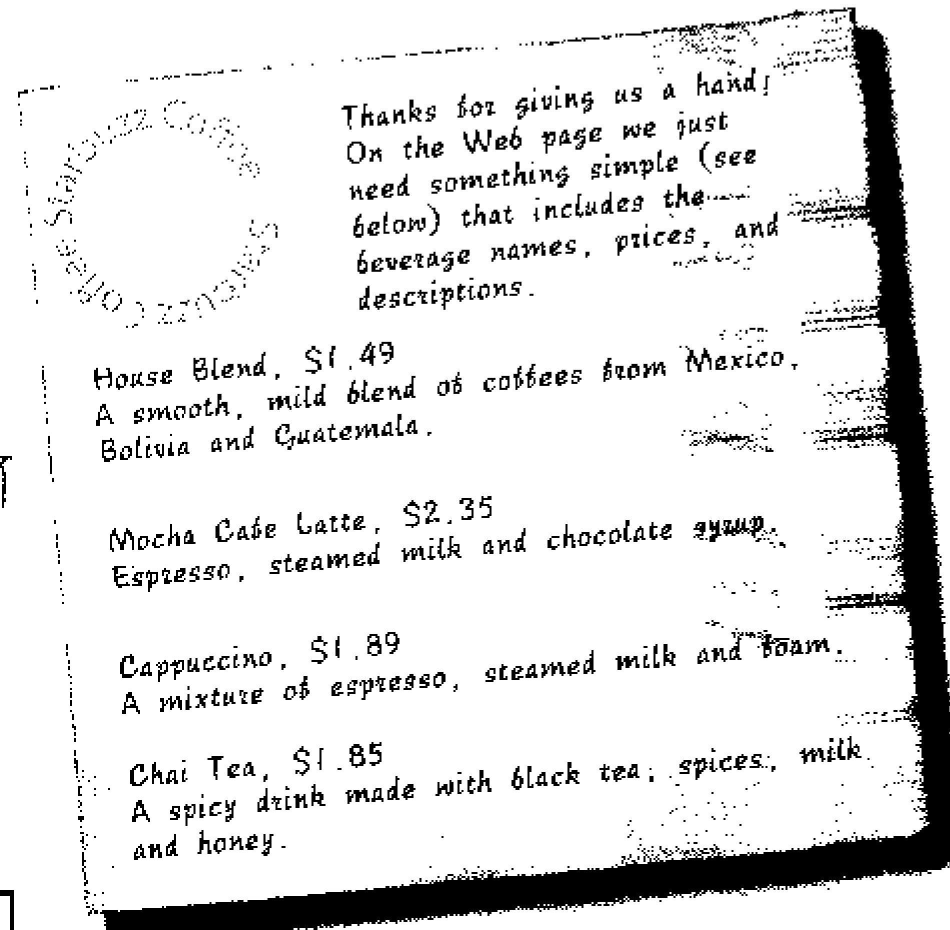


BRAIN POWER

从下面选项中选择你优先做的（单选）：

- ❑ A. 给小狗洗个澡
- ❑ B. 研究如何让自己的收支平衡
- ❑ C. 接受Starbuzz咖啡馆总裁的网站建设工作
- ❑ D. 安排与牙医会面。





Sharpen your pencil

看看餐巾纸上写了些什么，你可以把大致结构确定下来吗？换句话说，有明确的标题、段落了吗？是不是缺少某些东西（比如：一个标题）？

给餐巾纸上的东西（用铅笔）标记结构，并添加缺少的部分。

答案在本章末尾。

* 如果你在前一页选择了A、B或者D，我们建议你把这本书捐赠到优秀的图书馆去吧，或者冬天用来生火，更好的办法是到亚马逊网站上卖掉它，捞点现金回来。

建立Starbuzz网页

当然，唯一的问题就是你还没有实际创建过任何网页。不过，这也是你专心学习HTML的原因，对不对？

别担心，以下是后面几页你要做的：

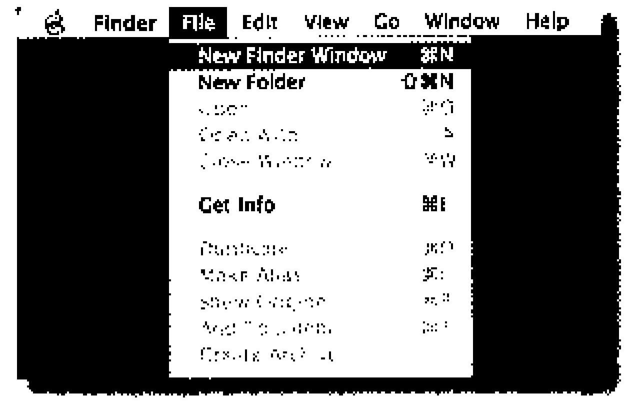
1. 用你喜欢的文本编辑器建立一个HTML文档。
2. 把Starbuzz CEO写给你的纸巾上的内容打进去。
3. 以index.html为名保存。
4. 用你喜欢的浏览器打开index.html文件，然后，等待奇迹的发生。



创建一个HTML文件（使用Mac系统）

所有的HTML文件都是文本文件。你需要一个允许创建纯文本的应用程序，不需要特殊的格式或者字体。你需要的只是无格式的纯文本。

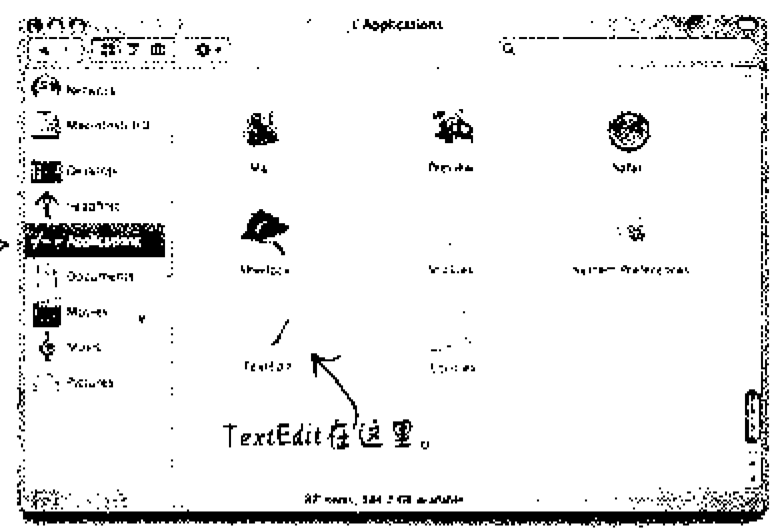
在本书中，我们将在Mac系统上使用TextEdit，当然，选择其他文本编辑器也一样能完成。如果你使用的工具是Windows的话，就可以跳过这几页直接到Windows部分。



第一步：

打开Applications文件夹

TextEdit应用程序在Applications文件夹里。最简单的方法是在File菜单中选择“New Finder Window”并以快捷方式搜寻Applications。找到后单击Applications。



你在这里找。

第二步：

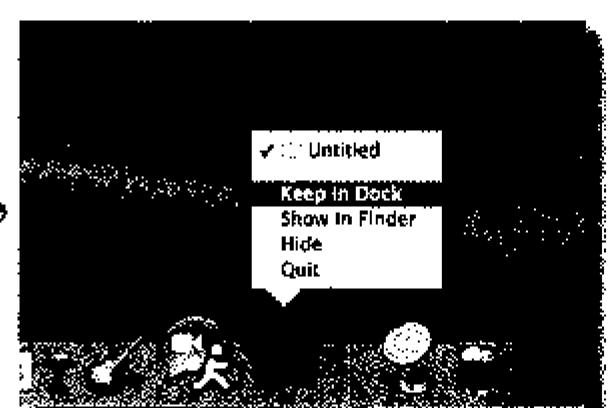
找到并运行TextEdit

在Applications文件夹中可能有一大堆应用程序，这时往下翻直至看到TextEdit。双击TextEdit图标运行它。

第三步（可选的）：

让TextEdit保留在Dock中

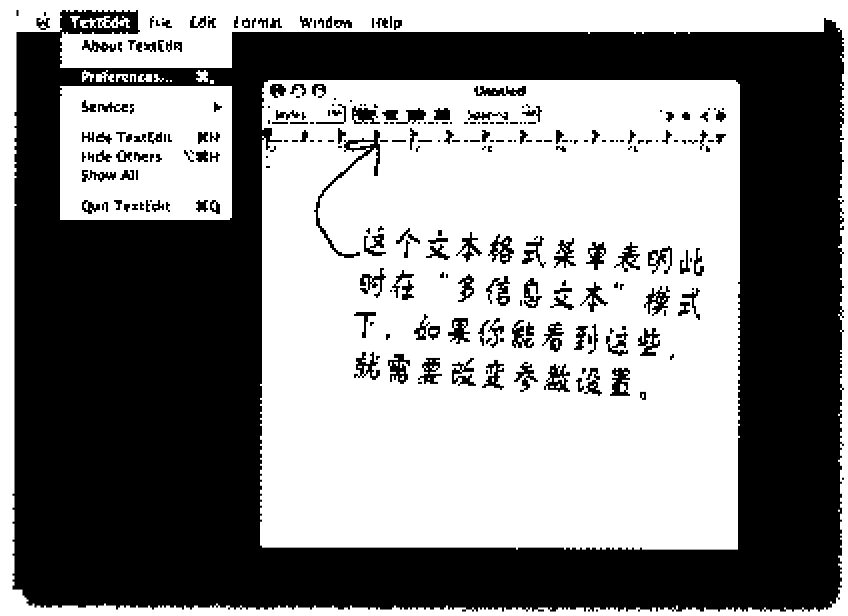
如果你想更便捷的话，单击TextEdit图标使之出现在Dock中（一旦应用程序启动，图标就会出现），在弹出菜单中选择“Keep in Dock”选项。TextEdit图标会出现在Dock中，以后每次运行时就不用再打开Applications文件夹了。



第四步：

修改TextEdit的参数设置

默认的TextEdit是“多信息文本”模式，这意味着保存时它会添加它独有的格式和特殊字符，而这不是你想要的。所以你必须修改TextEdit的选项，保证所保存的是纯文本。在TextEdit菜单中选择“Preferences”选项。



第五步：

把Preferences设置为纯文本

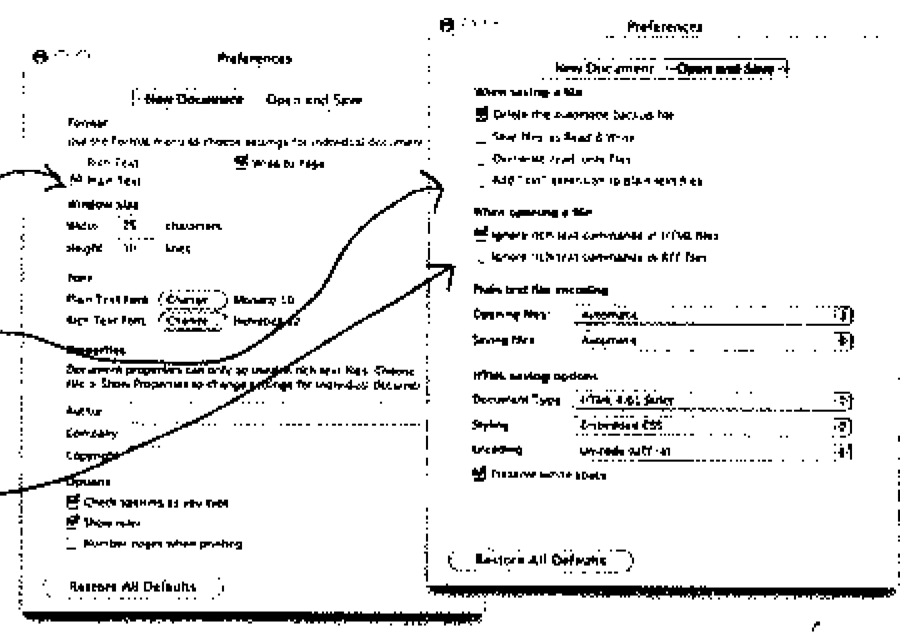
一旦你看到Preferences对话框，你需要做下面三件事情。

首先，在“New Document”选项卡中选择“Plain text”作为默认编辑器模式。

然后，在“Open and Save”选项卡中，确定没有勾选“Add.txt extension to plain text files”选项。

最后，确定勾选“Ignore rich text commands in HTML files”选项。

好了，单击左上角的红色按钮关闭对话框。



第六步：

退出并重启

在TextEdit菜单中选择Quit退出TextEdit。接着重启程序，这次，你将看到窗口顶部没有复杂的文本格式菜单了。那你就做好编写HTML的准备了。



创建一个HTML文件（使用Windows系统）

如果你使用的是Windows XP，那就得阅读这几页，如果不是，你可以跳过这几页。或者，你想坐到后排而不打算发问的话，我们也没意见。

或者Windows的其他版本。

在XP中我们使用记事本（Notepad）来创建HTML文件，记事本在每个Windows系统中都是捆绑的，价格合理，也易于使用。如果你有自己喜欢的在XP下运行的编辑器，这也是可以的，只要确保能够创建以“.html”为扩展名的纯文本文件即可。

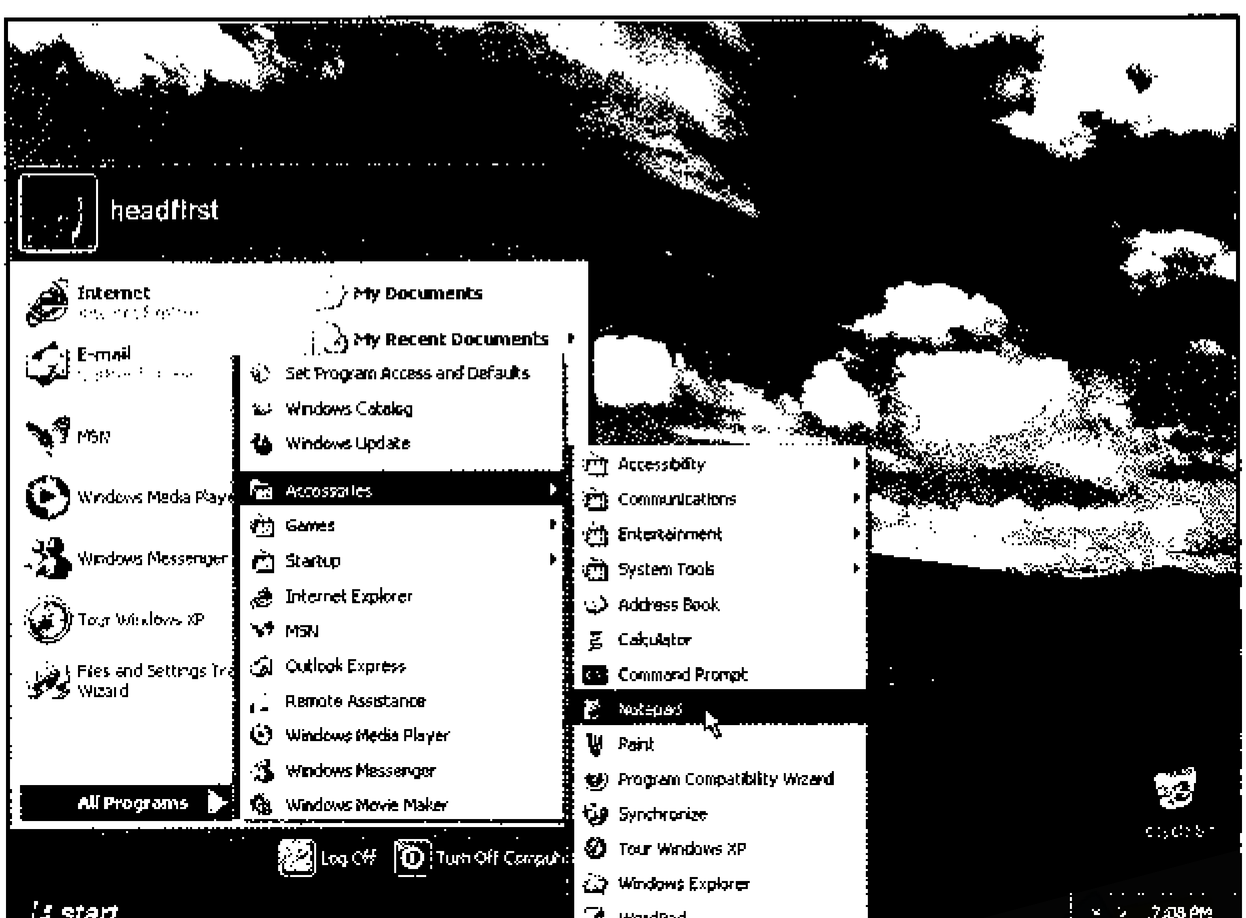
如果你用的是Windows的其他版本，也能以同样的方式找到记事本。

假设你使用的是记事本，现在看看怎么创建你的第一个HTML文件：

第一步：

打开“开始”菜单并找到记事本

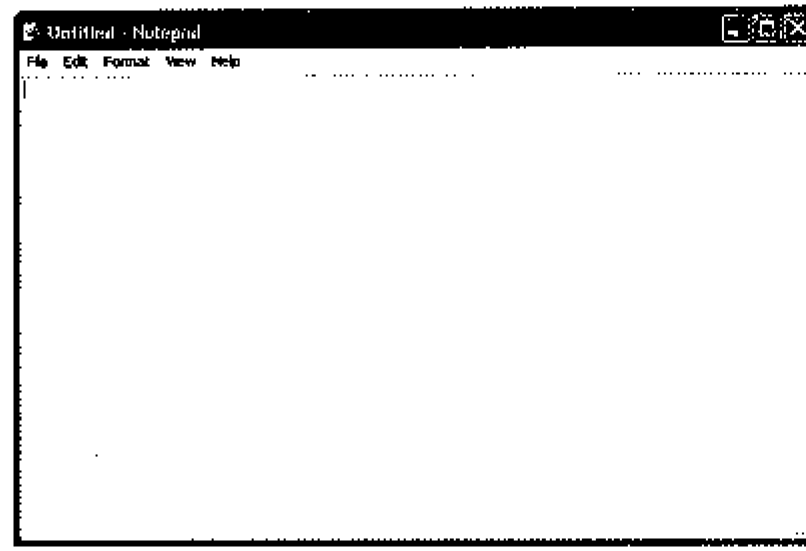
你会在附件中找到记事本，最简单的方法是单击“开始”按钮，选择“所有程序/附件/记事本”选项。



第二步：

打开记事本

一旦你在附件中选择了记事本，点击它将会打开一个空白的窗口，你就可以在其中编写HTML了。



但是推荐使用。

第三步（可选的）：

不要隐藏常见文件类型的文件扩展名。

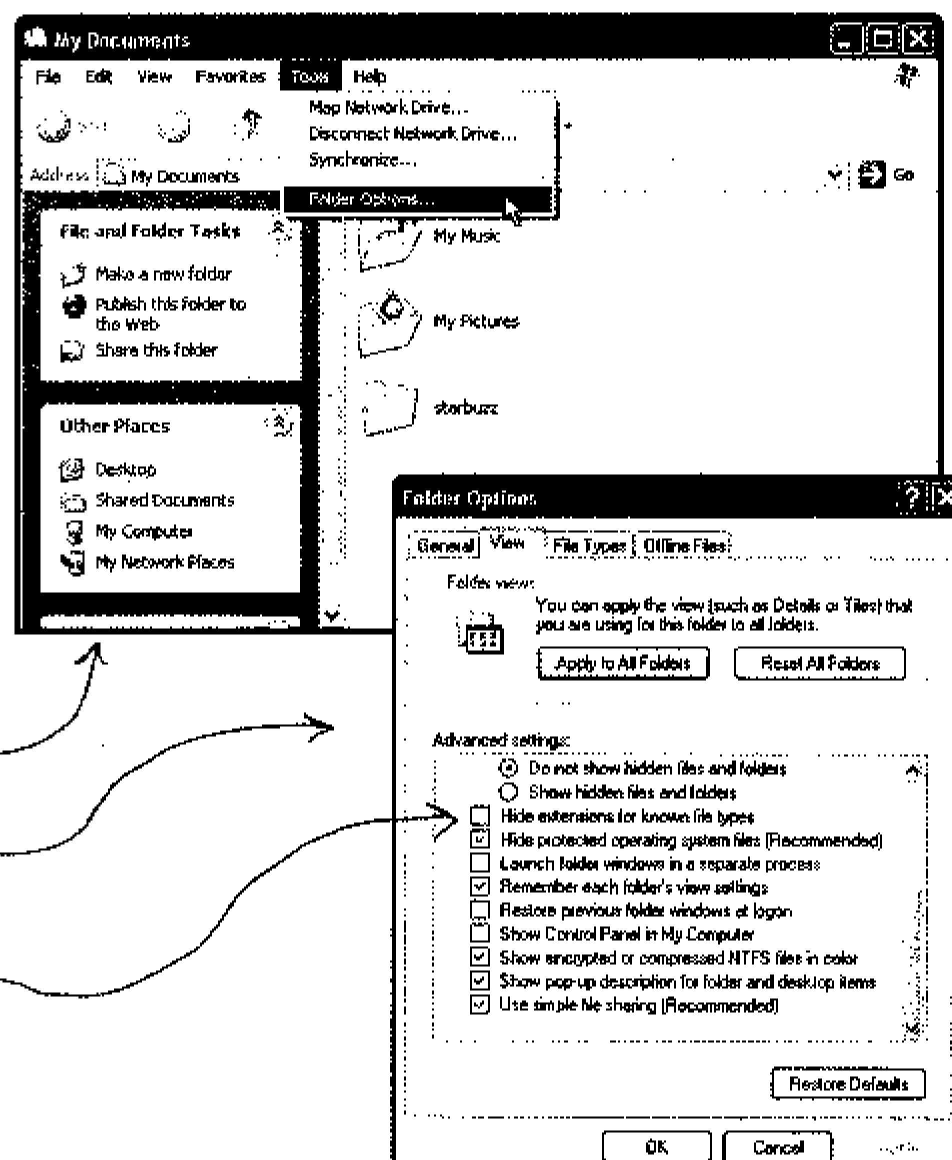
在XP资源管理器中，默认情况下常见文件类型的文件扩展名会被隐藏。例如，一个名叫Irule.html的文件在资源管理器中显示是“Irule”，而没有“.html”扩展名。

如果XP显示文件扩展名的话将减少许多混乱。因此，我们需要修改文件夹选项使文件扩展名可见。

首先，在资源管理器窗口中的工具菜单中选择“文件夹选项”命令。

接着，选择“查看”选项卡，在“高级设置”中，往下找到“隐藏已知文件类型的扩展名”，确保没有勾选它。

好了，单击“确定”按钮保存设置，这样就可以在浏览器中看到文件的扩展名了。



there are no Dumb Questions

问：为什么我要用简单的文体编辑器？不是还有诸如Dearmweaver、FrontPage和Golive等功能更强大的网页设计工具吗？

答： 你阅读本书的原因是因为你想了解构建网页的技术，对不对？你所说的那些都是些功能强大的工具，但是它们为你做的太多了，除非你已经是个HTML和CSS的高手，否则还没有使用它们的必要。

不过，一旦你精通此道了，这些工具确实能提供诸如语法检查和预览等有用功能。此时，当你浏览“代码”窗口时，你会看懂其中的所有内容，你还将发现直接修改原始HTML和CSS代码比通过用户界面来修改要快得多。同时，你也会发现当标准改变时，这些工具并不能及时更新，而且还可能直到发布下一个版本才支持最新的标准。由于你知道不通过工具就能更新HTML和CSS的方式，因而总是能够跟上最新最好的标准。

问：我有编辑器了，但是我应该使用哪个浏览器呢？IE、Firefox、Opera、Safari，太多选择了，怎么办？

答： 最简单的回答是：根据自己的爱好选择。HTML和CSS是基于工业标准的，这意味着所有的浏览器都用同样方式支持HTML和CSS（要做到支持性最好，你要确定你使用的浏览器是最新版本）。

最复杂的回答是：事实上，用上述浏览器处理网页的方式有细微的差别。如果你想令各种浏览器的使用者都能访问你的网页，就得用多种浏览器测试。有些网页在各种浏览器中看起来都一样，有些则不然。当你学习越多的HTML和CSS，这些细微的差别对你的影响就会越大，我们会在整本书中贯穿这些微妙之处。

如果你想找个好的浏览器，试试Mozilla的Firefox吧，它对HTML和CSS都有很好的支持。

问：我在电脑上创建了这些文件，如何在因特网上浏览它们呢？

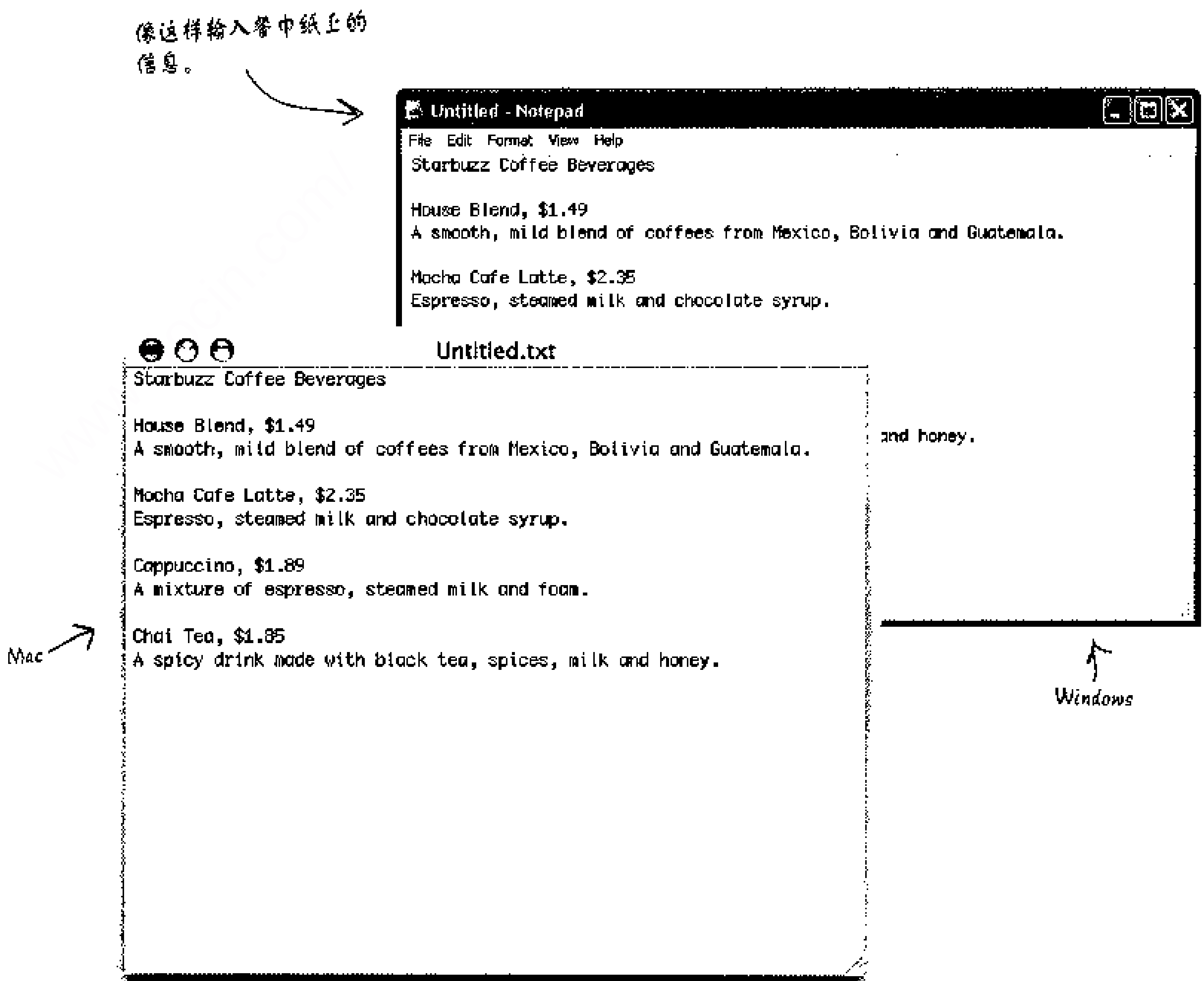
答： 能在自己的电脑上创建并调试HTML文件，然后发布到网上，这是令人兴奋的工作。现在我们关注的是如何创建文件和文件的内容。稍后我们会研究如何在网上发布。

现在，让我们回到Starbuzz咖啡馆的工作上来……



好了！你已经知道创建纯文本文件的基础，现在你要做的是把内容录入到你的文件中并保存，然后把它加载到你的浏览器。

从录入CEO给你的餐巾纸上的饮料开始：这些饮料是你网页的内容。你要在内容中加入HTML标记来使它有点结构，但现在，先把基本的内容输入电脑。开始的时候，把“Starbuzz Coffee Beverages”添加到文件开头。



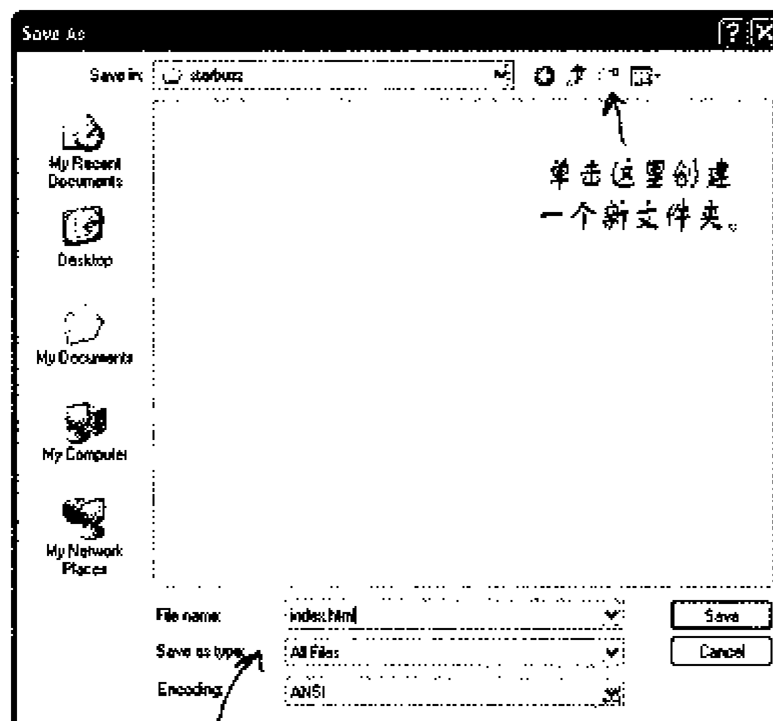
保存你的成果……

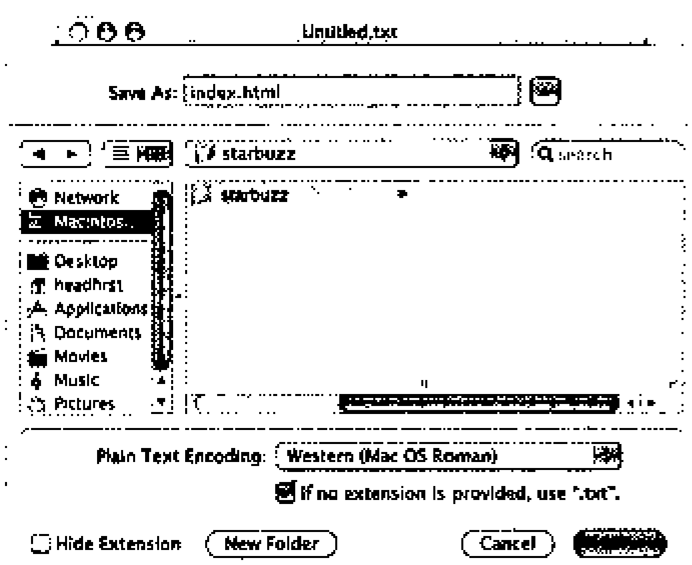
当你把餐巾纸上的内容输入完成后，把你的成果以“index.html”为文件名保存。在此之前，你应该建立一个名为“starbuzz”的文件夹保存站点文件。

做完这些预备工作后，在“文件”菜单中选择“保存”命令，这时就会弹出“另存为”对话框。然后，以下是你要做的：

Windows

① 首先，创建一个叫“starbuzz”的文件夹来保存与此有关的所有文件。单击“新建文件夹”按钮。





Mac

Windows用户还需要选择“所有文件”作为文件类型，否则记事本会在文件名后添加“.txt”。

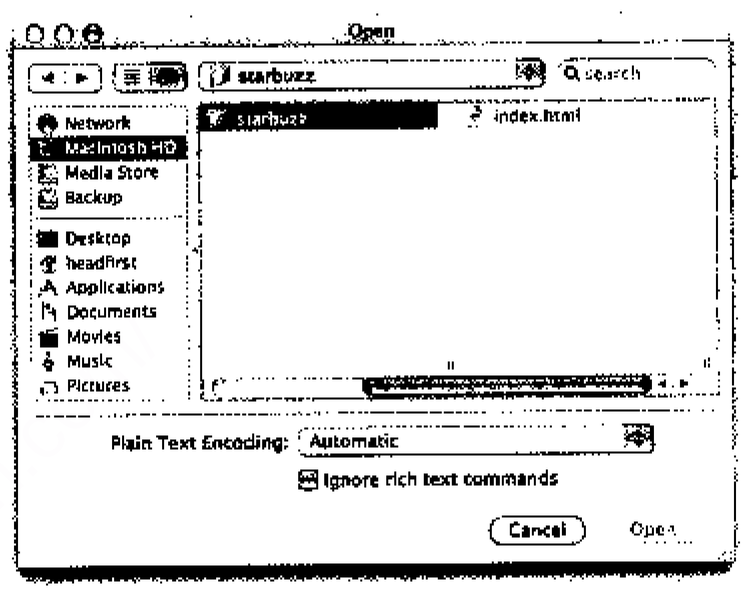
单击这里创建一个新文件夹。

② 第二步，单击新建的“starbuzz”文件夹图标，将“index.html”作为文件名输入，再单击“保存”按钮。

用浏览器打开你的网页

准备打开你的第一个网页了吗？用你喜欢的浏览器，从“文件”菜单下选择“打开文件”命令（Windows XP和IE用户选择“打开”命令），并找到“index.html”文件，选中它，并单击“打开”按钮。

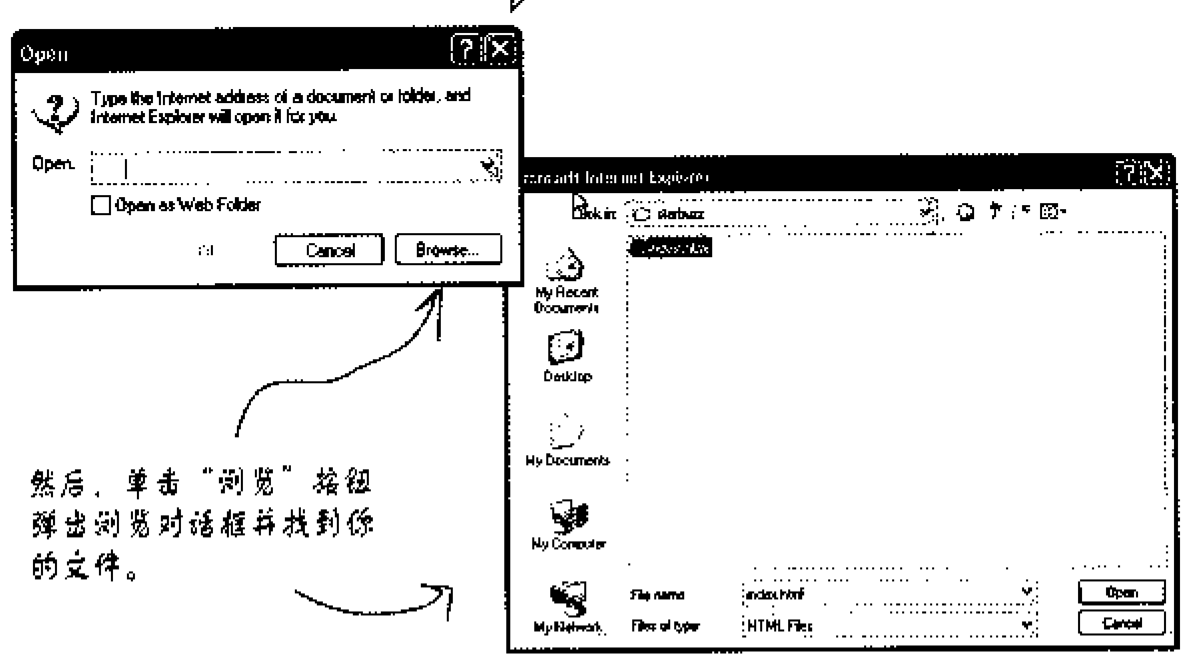
Mac



在Mac系统中，找到“index.html”文件，单击此文件图标选中它，然后单击“打开”按钮。

在Windows IE中需要两步，首先是打开“打开”对话框。

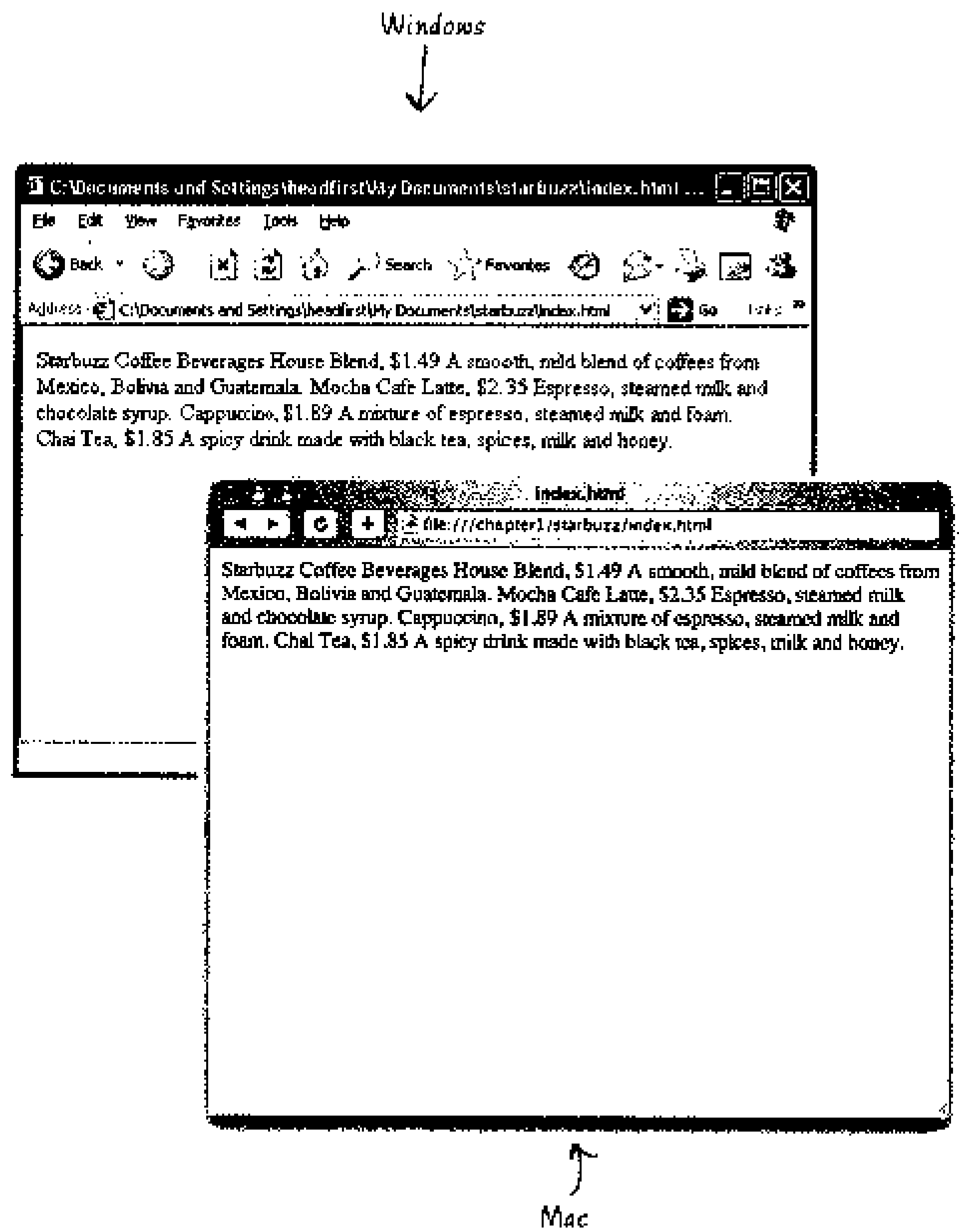
Windows



然后，单击“浏览”按钮弹出浏览对话框并找到你的文件。

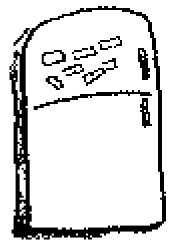
调试你的网页……

成功了！你会发现浏览器加载了网页，尽管看起来还不是很令人满意。但这只不过是因为你所做的工作仅局限在创建一个网页并用浏览器浏览它。到现在为止，你只是输入了网页的内容。这时就该HTML发挥作用了。HTML为你提供了一种告诉浏览器网页的结构的方式。什么是结构？如你所见，就是一种标记文本的手段，令浏览器知道哪里是标题，哪里是主体，哪里是子标题等。一旦浏览器知道了一些有关结构的信息，它就会以更具意义和可读性的方式显示页面。





标记帖



好了，接下来让我们添加结构……

你的工作就是给从Starbuzz餐巾纸上得到的文本添加结构。用本页底部的那些冰箱帖（就像我们常用的便笺）上进行标记，以此来显示哪里是标题、子标题和段落文本。我们已经做了一些示范。你不需要用到下面所有的冰箱帖来完成工作，有些会用不上。

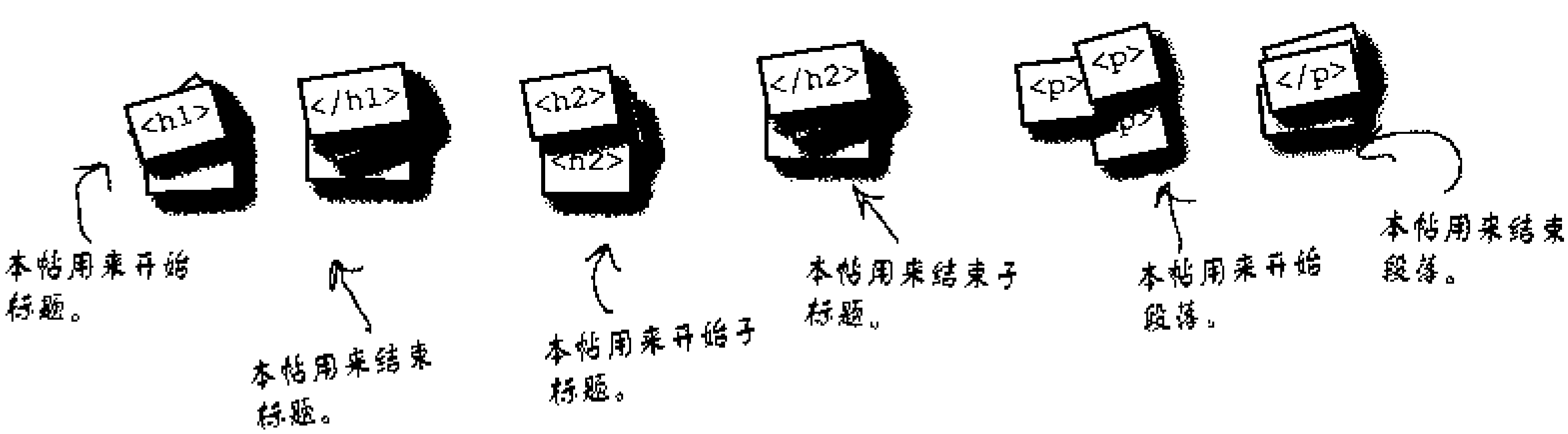
<h1>Starbuzz Coffee Beverages </h1>

House Blend, $1.49
A smooth, mild blend of coffees from Mexico, Bolivia
and Guatemala.

<h2>Mocha Cafe Latte, $2.35 </h2>
<p>Espresso, steamed milk and chocolate syrup. </p>

Cappuccino, $1.89
A mixture of espresso, steamed milk and foam.

Chai Tea, $1.85
A spicy drink made with black tea, spices, milk and
honey.





恭喜你，你已经写好了你的第一个HTML！

它们看起来很像冰箱帖，但是你的确已经用HTML标记了文本。你要知道，我们通常把这些冰箱帖称为标记。检查下面的标记并把它和上一页的冰箱帖进行比较。

用<h1>和</h1>标记标识标题。在这两个标记之间的所有文本就是标题的实际内容。

```
<h1> Starbuzz Coffee Beverages</h1>

<h2>House Blend, $1.49</h2>
<p> A smooth, mild blend of coffees from Mexico,
Bolivia and Guatemala.</p>

<h2> Mocha Cafe Latte, $2.35</h2>
<p> Espresso, steamed milk and chocolate syrup.</p>

<h2> Cappuccino, $1.89</h2>
<p> A mixture of espresso, steamed milk and foam.</p>

<h2> Chai Tea, $1.85</h2>
<p> A spicy drink made with black tea, spices, milk
and honey. </p>
```

<h2>和</h2>包围着子标题。把<h2>标题当成<h1>标题的子标题。

<p>和</p>包围着一大块作为段落的文本，可以是一个句子或者很多句子。

注意不必把匹配的标记放在同一行。可以在它们之间任意放置内容。

我们完成了没有？

你已经标记好一个HTML文件了，这样就做好了一个网页吗？差不多。你已经见过<html>、<head>、<title>和<body>等标记，我们现在要做的不过是把它们添加进去，使你的网页成为一流的……

首先，添加<html></html>标记以包围你的HTML。用此通知浏览器文件的内容是HTML。

然后，添加<head>、</head>标记。这两个头部标记内包含了你网页的信息，例如，标题。现在，可以这样看待它：你可以通过头部告诉浏览器有关网页的信息。

接下来把标题嵌入头部。通常标题会出现在浏览器窗口的顶部。

```
<html>
    <head>
        <title>Starbuzz Coffee</title>
    </head>

    <body>
        <h1>Starbuzz Coffee Beverages</h1>
        <h2>House Blend, $1.49</h2>
        <p>A smooth, mild blend of coffees from Mexico,
            Bolivia and Guatemala.</p>

        <h2>Mocha Cafe Latte, $2.35</h2>
        <p>Espresso, steamed milk and chocolate syrup.</p>

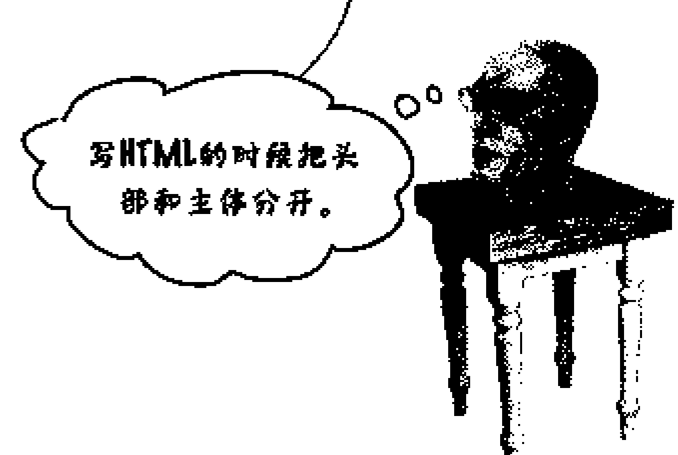
        <h2>Cappuccino, $1.89</h2>
        <p>A mixture of espresso, steamed milk and foam.</p>

        <h2>Chai Tea, $1.85</h2>
        <p>A spicy drink made with black tea, spices,
            milk and honey.</p>
    </body>
</html>
```

头部由<head></head>标记及它们之间的内容组成。

主体由<body></body>标记及它们之间的内容组成。

主体包含了网页的所有内容和结构——你在浏览器中看到的那些部分。

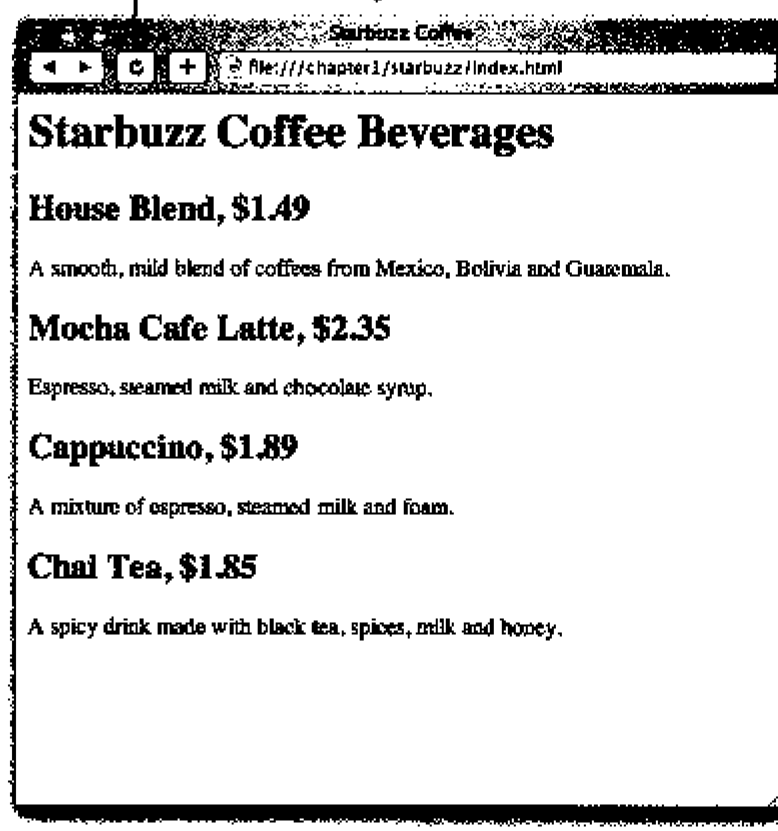


另一个测试……

继续工作，在你的“index.html”文件中添加<head>、</head>、<title>、</title>、<body>、</body>标记。完成后，保存，并用浏览器重新加载文件。

注意，你在<head>元素中指定的标题出现在这里。

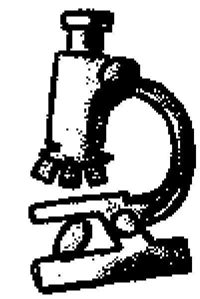
可以再次选择“打开文件”菜单项或者直接使用浏览器的重载按钮重新载入index.html文件。



现在看起来好一点了。浏览器解析标记并以此显示网页，页面不但更结构化，而且可读性也更好。



解剖标记……



好，你已经见过不少标记了，现在我们仔细研究一下标记是如何工作的……

通常在部分内容周围添加标记。这里我们使用标记告诉浏览器"Starbuzz Coffee Beverages"是最高级标题（即一级标题）。

这里是用来开始标题的开始标记。

这个是用来结束标题的结束标记。在这里，</h1>用来结束<h1>标题。它之所以是个结束标记，因为它出现在内容后边，而且在"h1"之前还有"/"。所有的结束标记都有"/"。

```
<h1> Starbuzz Coffee Beverages </h1>
```

标记通常由尖括号及标记名组成，尖括号即"<"和">"。

整个语句称为一个元素，在这里我们称其为<h1>元素。一个元素由开始标记、结束标记及它们之间的内容组成。

我们称一对开始标记和结束标记为匹配标记。

用包围文本的成对标记告诉浏览器网页的结构。

记住：

元素＝开始标记＋内容＋结束标记

there are no Dumb Questions

问： **匹配标记不需要放在同一行？**

答： 不需要。记住：浏览器并不介意换行、回车和大多数空格。因此，标记可以在同一行或者不同行的任意地方开始和结束。你只需确保是以开始标记开始（如<h2>），以结束标记结束（如</h2>）就行了。

问： **为什么结束标记必须有“/”符号？**

答： 结束标记中的“/”符号使你和浏览器明白一段特定的内容在哪儿结束。否则，结束标记看起来就像开始标记了，对不对？

问： **我发现某些页中的HTML有时没有用来匹配开始标记的结束标记。**

答： 标记应该都是匹配的。通常来说，就算输入有错，浏览器还是会恰当地辨别出你的意思。但是就像你会看到的一样，现在无误的HTML会大有好处。如果你担心自己永远写不出完美的HTML，不要急，有许多工具能在你把网页发布到网站上之前更正你的编码。现在，你只需要养成书写匹配的开始标记和结束标记的习惯就行了。

问： **在休闲室例子中的<img src=“drinks.gif”>是怎么回事啊？是不是忘记写结束标记了啊？**

答： 哦，好眼力！有一些只用一个标记作为速记符号的元素。现在暂时不要去想它，我们会在下一章再提起。

问： **一个元素就是开始标记＋内容＋结束标记，不能在其他标记中嵌入标记吗？就像头部和主体都在<html>标记内？**

答： 是的，HTML标记经常这样嵌套。其实你思考一下便知道，每个HTML有一个主体，主体又包含了一个段落等，这是很自然，很正常的。许多HTML元素会在它们的标记之间嵌入另外一个元素。我们将在本书的以后章节好好研究这类问题，现在你只需注意网页中的元素是怎么互相关联的就可以了。

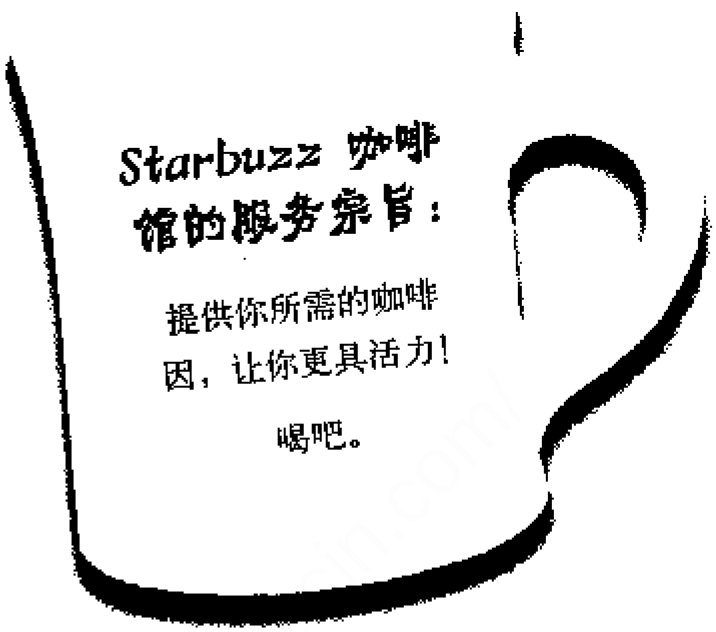
BRAIN POWER

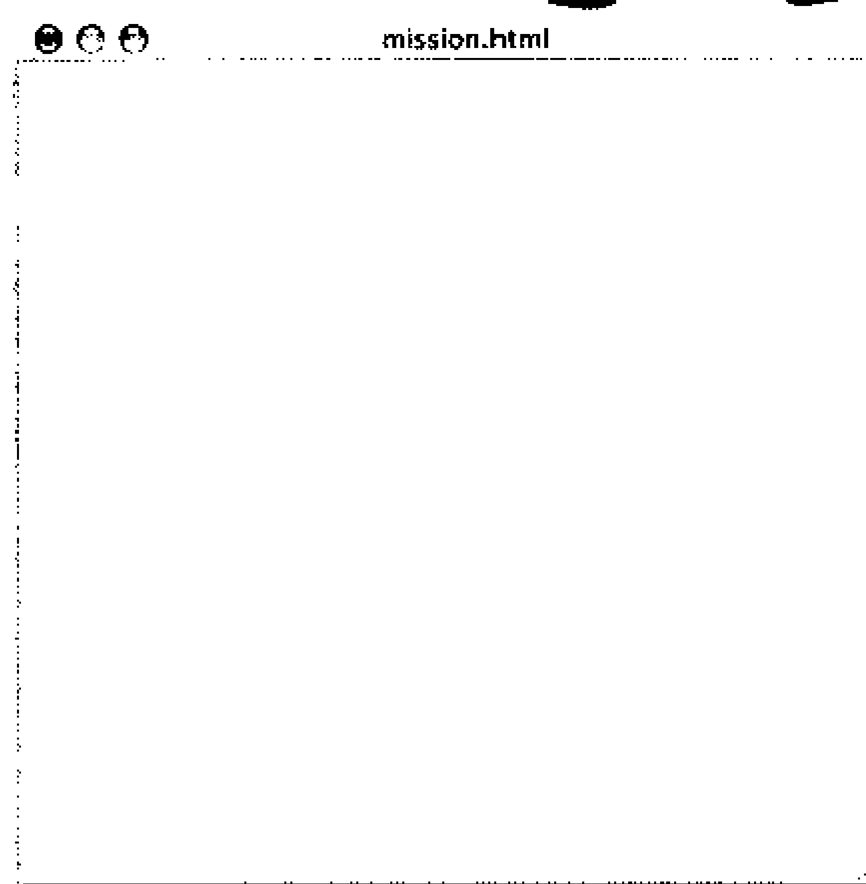
标记其实比你迄今所见的会更有趣。这里有个段落标记，还添加了一些其他内容，猜猜它们起什么作用？

```
<p id="houseblend">A smooth, mild blend
of coffees from Mexico, Bolivia and
Guatemala.</p>
```

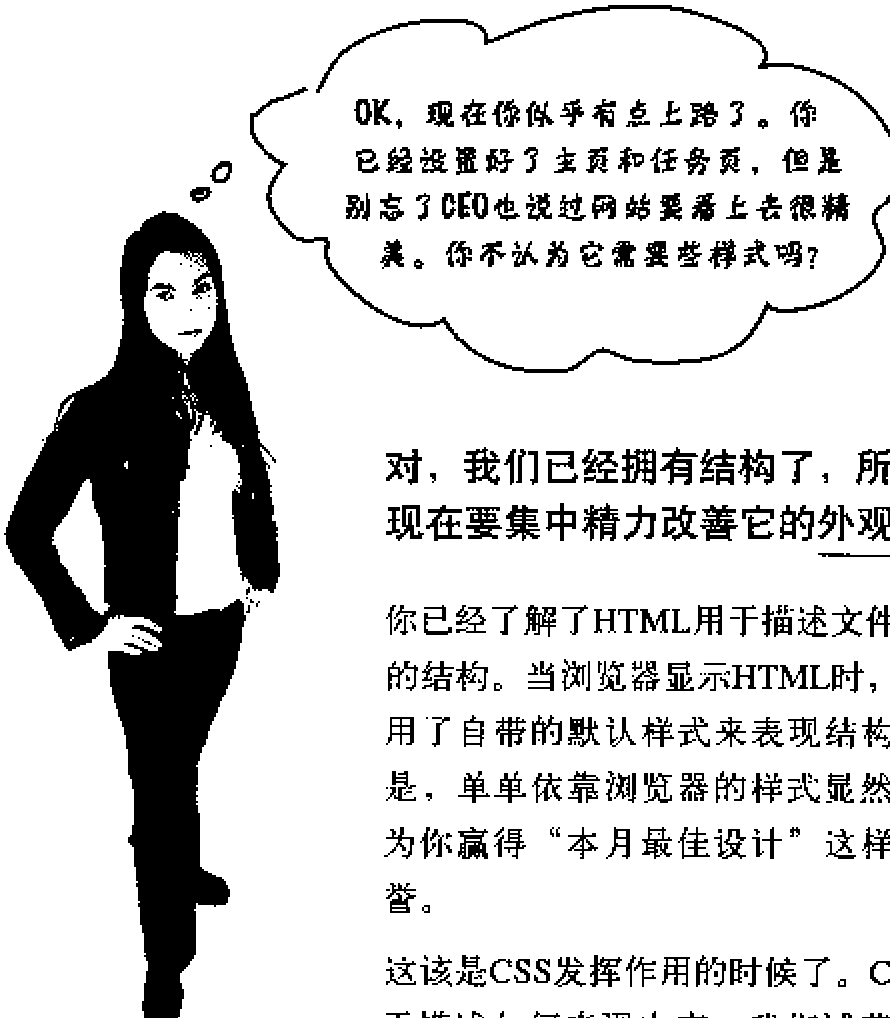
练习







1. 在这儿为新的“mission.html”页编写HTML。
2. 用文本编辑器把HTML输入进去，并保存为“mission.html”，放到和“index.html”同一个文件夹下。
3. 做完以上工作后，用浏览器打开“mission.html”。
4. 在做下一步前检查你的工作……



对，我们已经拥有结构了，所以现在要集中精力改善它的外观。

你已经了解了HTML用于描述文件内容的结构。当浏览器显示HTML时，它使用了自带的默认样式来表现结构。但是，单单依靠浏览器的样式显然不能为你赢得“本月最佳设计”这样的赞誉。

这该是CSS发挥作用的时候了。CSS用于描述如何表现内容。我们试着来创建一些CSS以使Starbuzz网页看起来更美观（同时开创你的网站事业）。

CSS是级联样式表（Cascading Style Sheet）的缩写。我们以后再了解它的意思，现在只需要知道CSS为我们提供了方法，来告诉浏览器网页中的元素应该如何表现。

认识样式元素

为了加入样式，我们添加了一个新的元素到你的网页中——<style>元素。让我们回头看看Starbuzz的主页并添加些元素。来吧……

```
<html>
    <head>
        <title>Starbuzz Coffee</title>
        <style type="text/css">

        </style>
    </head>

    <body>
        <h1>Starbuzz Coffee Beverages</h1>

        <h2>House Blend, $1.49</h2>
        <p>A smooth, mild blend of coffees from Mexico, Bolivia and
            Guatemala.</p>

        <h2>Mocha Caffe Latte, $2.35</h2>
        <p>Espresso, steamed milk and chocolate syrup.</p>

        <h2>Cappuccino, $1.89</h2>
        <p>A mixture of espresso, steamed milk and milk foam.</p>

        <h2>Chai Tea, $1.85</h2>
        <p>A spicy drink made with black tea, spices, milk and honey.</p>
    </body>
</html>
```

<style>元素放置在HTML的头部中。

像其他元素一样，<style>元素有开始标记<style>和结束标记</style>……

但是，<style>标记还需要一个名为type的属性，用来告诉浏览器你所使用样式的种类。由于你即将使用CSS，因此指定为"text/css"类型。

在这里指定网页的样式。

there are no Dumb Questions

问： 元素可以拥有"属性"？这是什么意思？

答： 属性用来为一个元素提供附加信息。举例来说，如果有一个样式元素，属性允许你确切地描述这个元素。你会看到不同元素的更多属性；记住这些属性只是提供该元素的一些额外信息。

问： 为什么我必须指定样式类型"text/css"作为样式属性？还有别的什么样式吗？

答： 现在的浏览器没有什么其他的通用样式，但是那些HTML的设计者经常会预先考虑并预见到以后会有其他样式类型。我个人认为，我们应该屏住呼吸期待<style type="50sKitsch">的到来。

给Starbuzz添加样式……

既然HTML头部有了<style>元素，你只需提供些CSS，使得网页更加漂亮就可以了。接下来你会发现有些“CSS”已经为你准备好了。当你看到这个标志时，你会发现只需原样输入的HTML和CSS。相信我们，在看到它的功能之后，你就会学会标记是如何工作的。

看看以下这段CSS，然后把它添加到“index.html”文件中。完成输入后保存文件。



待烘烤 CSS

```
<html>
    <head>
        <title>Starbuzz Coffee</title>
        <style type="text/css">
            body {
                background-color: #d2b48c;
                margin-left: 20%;
                margin-right: 20%;
                border: 1px dotted gray;
                padding: 10px 10px 10px 10px;
                font-family: sans-serif;
            }
        </style>
    </head>

    <body>
        <h1>Starbuzz Coffee Beverages</h1>

        <h2>House Blend, $1.49</h2>
        <p>A smooth, mild blend of coffees from Mexico, Bolivia and
            Guatemala.</p>

        <h2>Mocha Caffe Latte, $2.35</h2>
        <p>Espresso, steamed milk and chocolate syrup.</p>

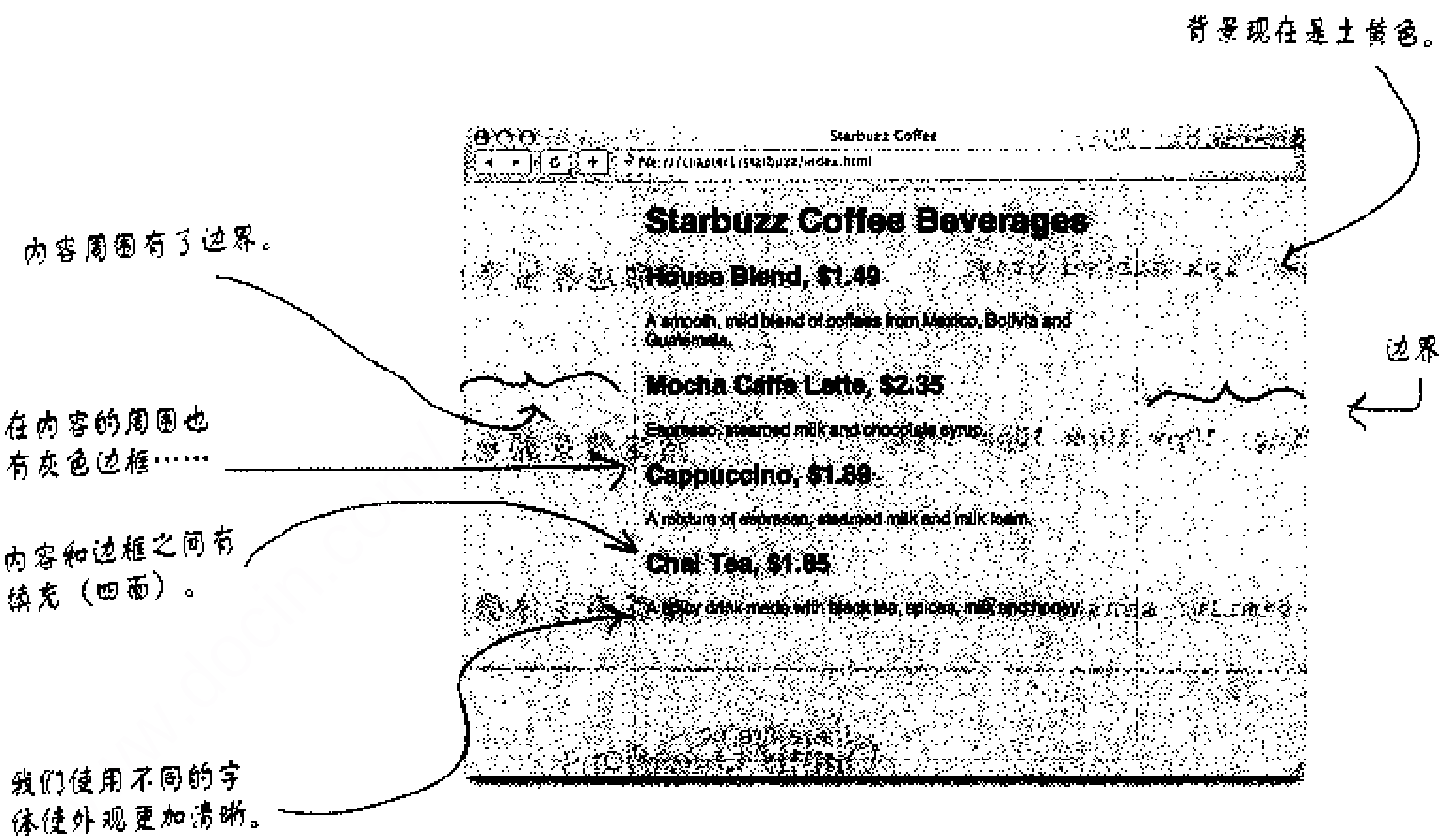
        <h2>Cappuccino, $1.89</h2>
        <p>A mixture of espresso, steamed milk and milk foam.</p>

        <h2>Chai Tea, $1.85</h2>
        <p>A spicy drink made with black tea, spices, milk and honey.</p>
    </body>
</html>
```

CSS使用一种和HTML截然不同的语法。

用样式冲浪……

该开始另一个测试了，再次重载你的“index.html”文件。这次你将看到Starbuzz网页拥有一个全新的面孔。





连连看

尽管你只瞥了一眼CSS，但已经看到它能干什么了。

将下面的样式定义同相应功能进行连线。

`background-color: #d2b48c;`	指定文本使用的字体。
`margin-left: 20%;` `margin-right: 20%;`	指定主体周围的边框线为点式且颜色为灰色。
`border: 1px dotted gray;`	左右边界设置为每边占页面的20%。
`padding: 10px 10px 10px 10px;`	指定背景颜色为土黄色。
`font-family: sans-serif;`	在页面主体周围创建某种填充。

there are no Dumb Questions

问： CSS看起来和HTML完全不同。为什么需要两种语言呢？那样我需要学更多的知识，对不对？

答： 你说CSS和HTML是两种完全不同的语言，这个观点很正确。因为它们做不同的工作。就像我们不能用英语平衡收支，不能用数学写诗一样，我们也不能用CSS创建结构或者用HTML创建样式，因为这不是当初设计它们的初衷。的确，这真是意味着你需要学习两门语言，但你会发现由于它们在各自的领域都十分出色，因此这比使用一种语言做两份工作来得轻松。

问： #d2b48c看起来不像一种颜色。为什么它是土黄色？

答： 用CSS有许多种方法来指定颜色。最普通的做法叫做“十六进制编码”，#d2b48c就是这种格式。它代表土黄色。现在别去考虑它，我们将在以后告诉你为什么它是土黄色。

问： 为什么CSS规则前边都有个“body”？它是什么意思？

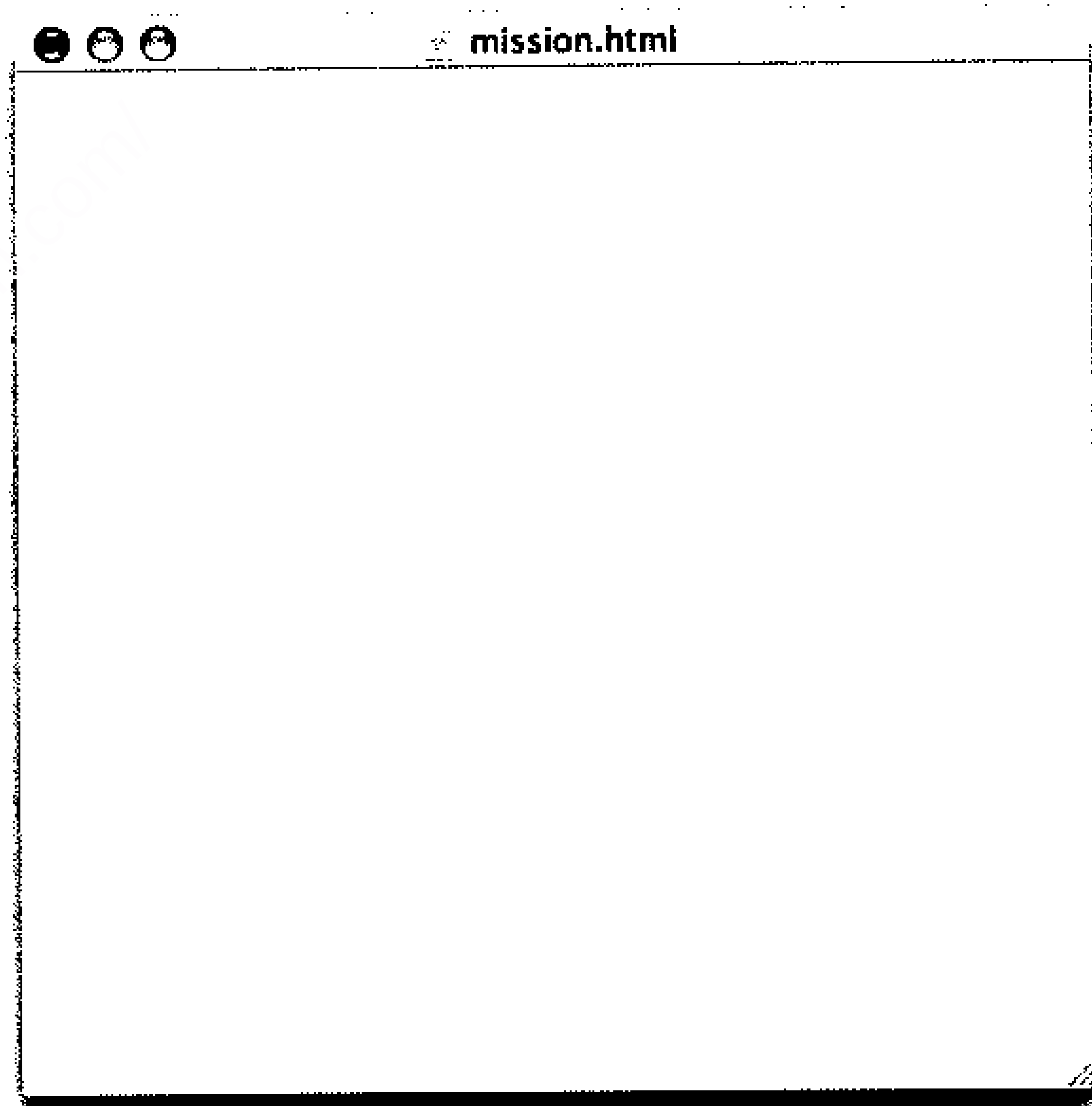
答： CSS中的“body”意味着“{”和“}”之间的所有CSS将应用于HTML<body>元素中的内容。因此，如果把字体设置为sans-serif，就表明网页主体的默认字体是sans-serif。

很快我们将开始讨论更多关于CSS工作原理的细节，所以你要继续读下去。很快你就会看到自己能熟练地应用大量的规则，做出更酷的设计。

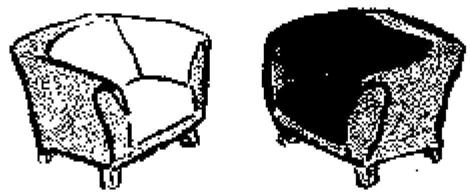
练习

现在你已经把一些样式加入到了Starbuzz “index.html” 网页中，接着使用相同的样式更新你的 “mission.html” 网页。

1. 用HTML写出下面的 “mission.html” 网页，然后添加新的CSS。
2. 用新的CSS升级你的 “mission.html” 网页。
3. 做完后，用浏览器重新载入 “mission.html” 。
4. 确保你的任务网页看起来就跟我们这章末尾的范例一样。



围炉夜话



今夜话题：HTML和CSS聊内容和样式

HTML	CSS
你好，CSS！非常高兴你在这里，因为我想澄清一下大家对于我们的混淆。	
	真地？什么样的混淆？
许多人认为我的标记告诉浏览器如何显示内容。不是这样的！我所有的工作都是关于结构的，而不是外观。	
	哎呀，我不想人们把我的工作全归功于你。
嗯，你可以看到有些人是如何变得稀里糊涂的；归根到底，这可能是因为只使用HTML不用CSS也能写出像样的网页。	
	“像样”这个词也许过火了点儿，你不觉得吗？我的意思是说，大多数浏览器直接显示HTML看起来过于蹩脚。人们需要认识到CSS是多么强大，并知道我能多么轻松地给他们的网页添加漂亮的样式。
嘿，我也很强大！使内容结构化比让它好看重要得多！样式太肤浅了，结构才是最重要的。	
	现实点儿！没有我你的网页是多么枯燥！不只这样，如果没有样式就没有人会把你的网页当回事。所有的东西看起来都是笨拙和不专业的。
喔～好自负！我想我不能从你那里期待别的什么了——你只是试图通过你一直鼓吹的样式发表流行声明。	

HTML	CSS
	流行声明？好的设计和布局会对网页的易读性和易用性起到巨大的功效。我灵活的样式规则允许设计者对你的元素做许多有趣的尝试，而不必同你的结构掺和在一起，你对此不满吗？
对，实际上我们是两种不同的语言，这很不错，因为我不希望样式设计者把你和我的结构元素掺和在一起。	
	不用担心，我们生活在两个不相关的领域里。
好，很明显对我而言，任何时候CSS讲的都是外语。	
	噢？HTML也能称为一门语言？谁见过有那么多标记，那么笨拙的东西？
数以百万计的网站编写人员不会同意你的说法。我有和内容配合得很好的清晰明了的句法构造。	
	看看CSS！它是如此简单优雅，没有愚笨的尖括号<围绕着><每个句子>。<看><我><能><像><HTML><先生><一样><说话><，><看着><我>!
嘿，傻瓜，有没有听说过结束标记？ 注意无论你在哪里，我都会用<style>标记包围你！祝你能够逃脱！	
	哈！我会做给你看的……因为，猜猜是什么？我就是能够逃脱！

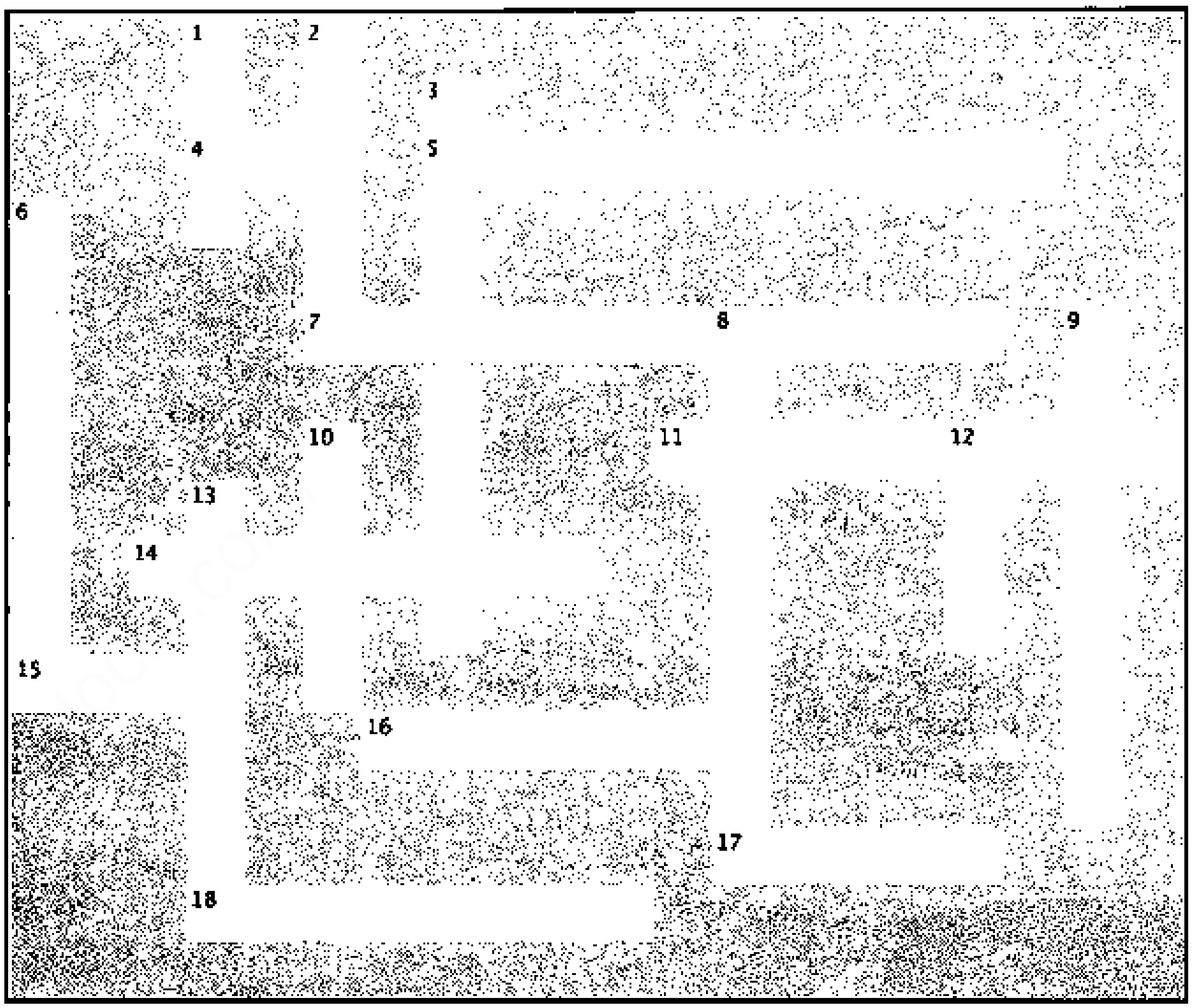


要点

- HTML和CSS是用来创建网页的语言。
- Web服务器存储并提供由HTML和CSS创建的网页。浏览器接收网页并基于HTML和CSS显示其中的内容。
- HTML是超文本标记语言（HyperText Markup Language）的缩写，用来结构化网页。
- CSS是级联样式表（Cascading Style Sheet）的缩写，用来控制HTML的外观。
- 我们使用HTML来标记内容，用标记提供结构。我们称匹配标记和它们之间的内容为元素。
- 一个元素由以下三个部分组成：一个开始标记、内容和一个结束标记。有些元素（如<img>）是例外。
- 开始标记可以拥有属性。我们看过一对了：type和align。
- 结束标记的左尖括号之后，标记名之前有一个“/”，用来区别于开始标记。
- 网页通常应该有一个<html>元素，并伴随有一个<head>元素和一个<body>元素。
- 网页的信息来源于<head>元素。
- 写进<body>元素的内容就是在浏览器中所能看见的东西。
- 浏览器忽略大多数的空白（制表符、回车、空格），但是可以用它们让你的HTML更具有可读性。
- 通过在<style>元素中输入CSS规则，给HTML网页添加CSS。<style>元素通常嵌在<head> 元素中。
- 在HTML中用CSS指定每个元素的样式特性。

HTML填字游戏

该让你大脑的左半球开始工作了。这是你的填字游戏，所填单词均来自本章。



横排提示：

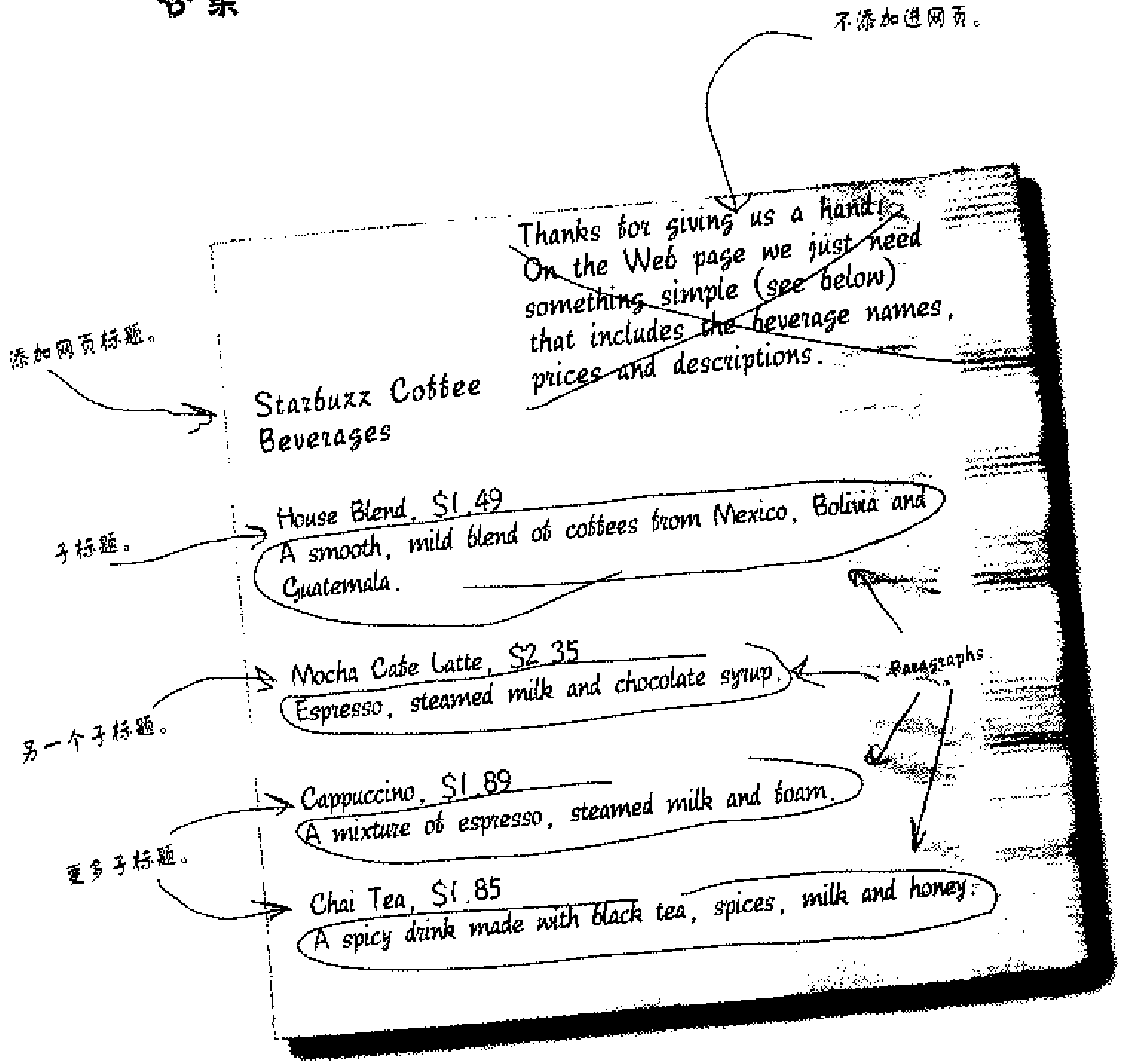
4. 我们强调的。
5. 在HTML中通常把它们分离。
7. 你要使用CSS进行控制的。
11. 你标记内容的目的。
14. 唯一存在的样式类型。
15. 有关网页的信息包括在其中。
16. 两个标记及内容。
17. 你用这个标记指定外观。
18. 启动网站事业的公司。

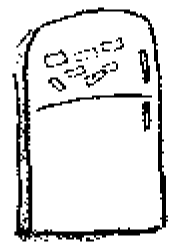
竖排提示：

1. 我们在网页上看到的。
2. HTML中的“M”。
3. 浏览器忽略的。
6. 我们期待的样式。
8. 提供附加信息的标记。
9. <p>元素的目的。
10. 浏览每个页面时浏览器顶部出现的。
12. 开始和结束。
13. 它们有六个。

Sharpen your pencil 答案

给餐巾纸上的东西标记结构（用铅笔），并添加缺少的部分。





标记帖解答

你的工作就是给从Starbuzz餐巾纸上得到的文本添加结构。用本页底部的那些冰箱帖(就像我们常用的便笺)上进行标记，以此来显示哪里是标题、子标题和段落文本。我们已经做了一些示范。你不需要用到下面所有的冰箱帖来完成工作，有些会用不上。

<h1> Starbuzz Coffee Beverages </h1>

<h2> House Blend, $1.49 </h2>
<p> A smooth, mild blend of coffees from Mexico, Bolivia and Guatemala. </p>

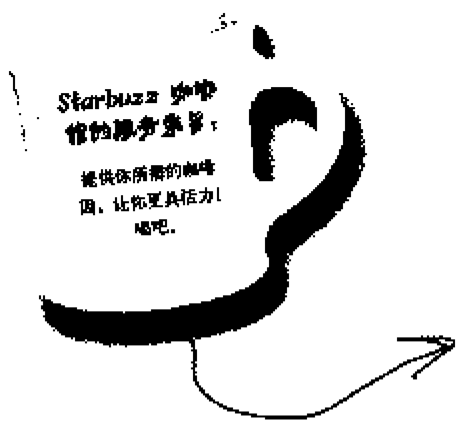
<h2> Mocha Cafe Latte, $2.35 </h2>
<p> Espresso, steamed milk and chocolate syrup. </p>

<h2> Cappuccino, $1.89 </h2>
<p> A mixture of espresso, steamed milk and foam. </p>

<h2> Chai Tea, $1.85 </h2>
<p> A spicy drink made with black tea, spices, milk and honey. </p>

<h1> </h1>

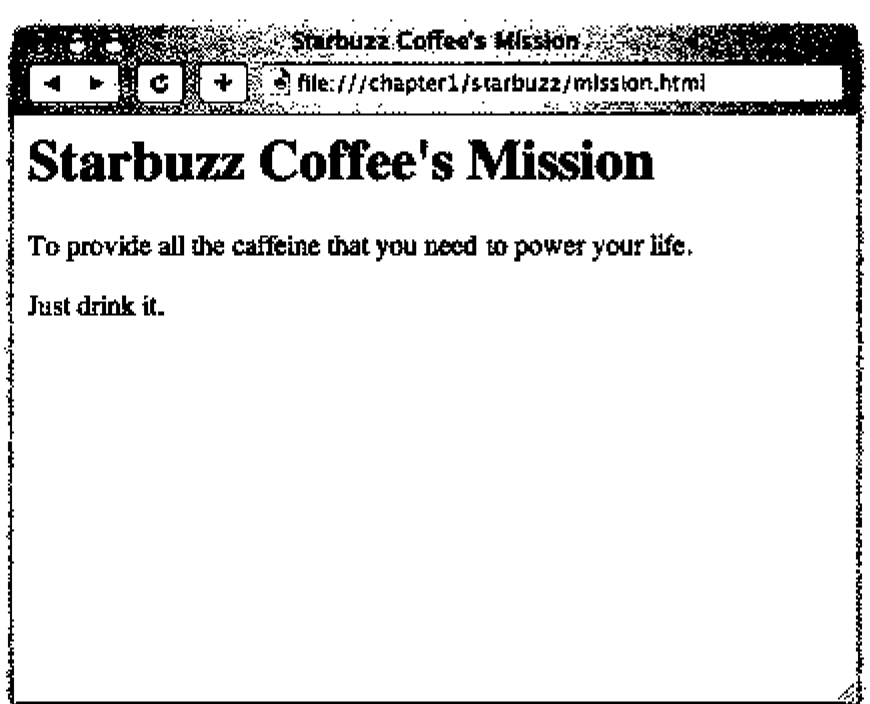




mission.html

```
<html>
  <head>
    <title>Starbuzz Coffee's Mission</title>
  </head>
  <body>
    <h1>Starbuzz Coffee's Mission</h1>
    <p>To provide all the caffeine that you need to
power your life.</p>
    <p>Just drink it.</p>
  </body>
</html>
```

这里是HTML。

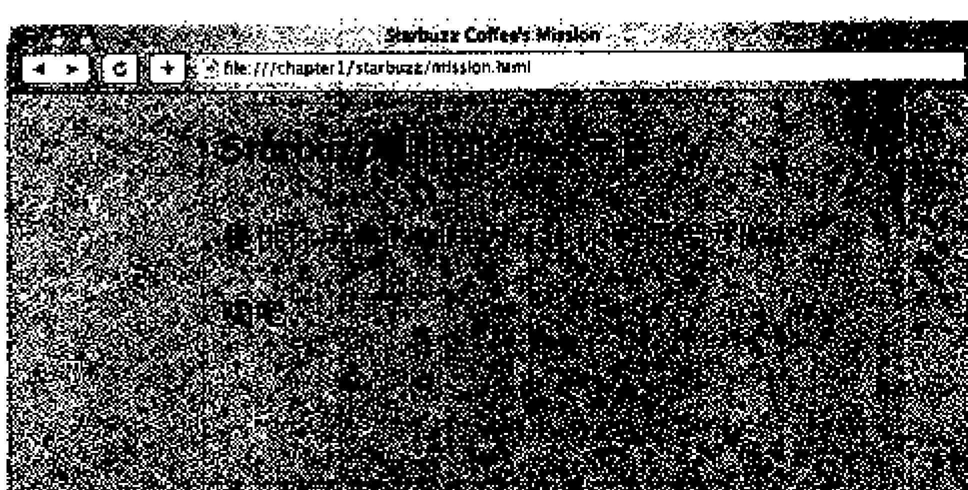


这里是HTML在浏览器中显示的结果。

练习答案

mission.html

```
<html>
  <head>
    <title>Starbuzz Coffee's Mission</title>
    <style type="text/css">
      body {
            background-color: #d2b48c;
            margin-left: 20%;
            margin-right: 20%;
            border: 1px dotted gray;
            padding: 10px 10px 10px 10px;
            font-family: sans-serif;
      }
  </style>
  </head>
  <body>
    <h1>Starbuzz Coffee's Mission</h1>
    <p>To provide all the caffeine that you need to power your life.</p>
    <p>Just drink it.</p>
  </body>
</html>
```



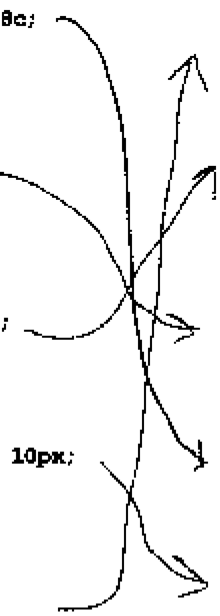
练习答案

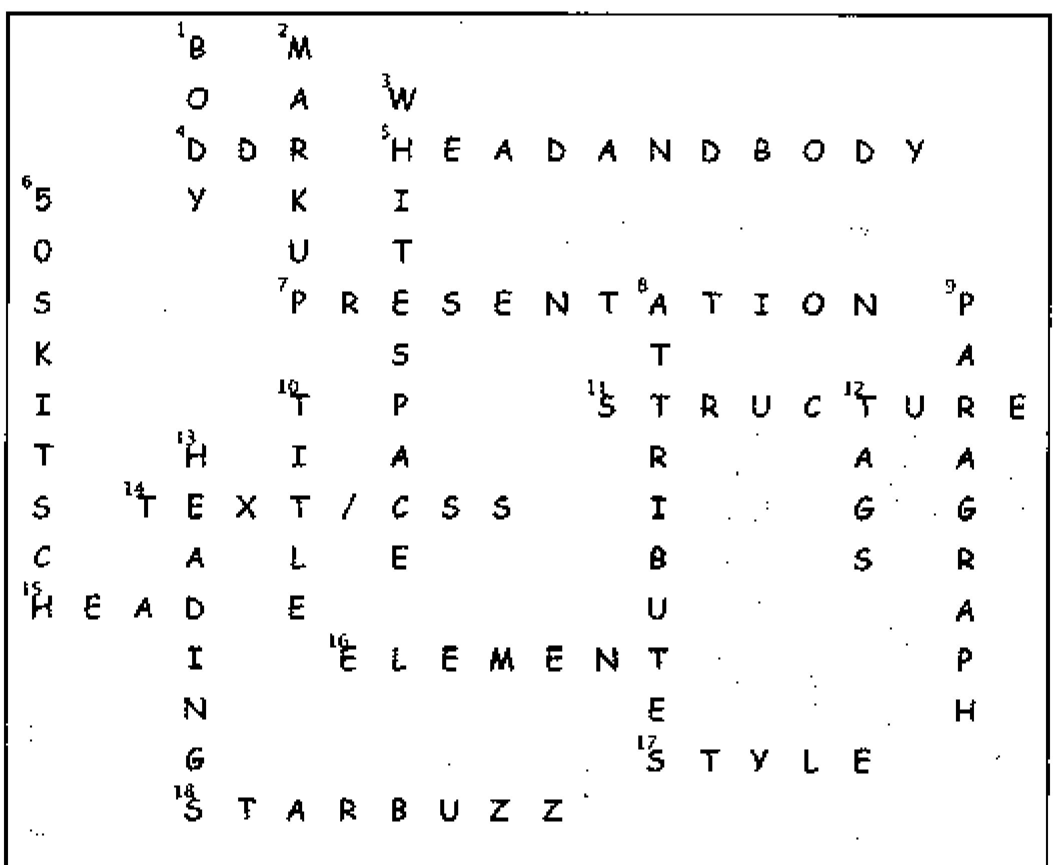
连连看

尽管你只瞥了一眼CSS，但已经看到它能干什么了。

将下面的样式定义同相应功能进行连线。

`background-color: #d2b48c;`	指定文本使用的字体。
`margin-left: 20%; margin-right: 20%;`	指定主体周围的边框线为点式且颜色为灰色。
`border: 1px dotted gray;`	左右边界设置为每边占页面的20%。
`padding: 10px 10px 10px 10px;`	指定背景颜色为土黄色。
`font-family: sans-serif;`	在页面主体周围创建某种填充。





2 深入理解超文本

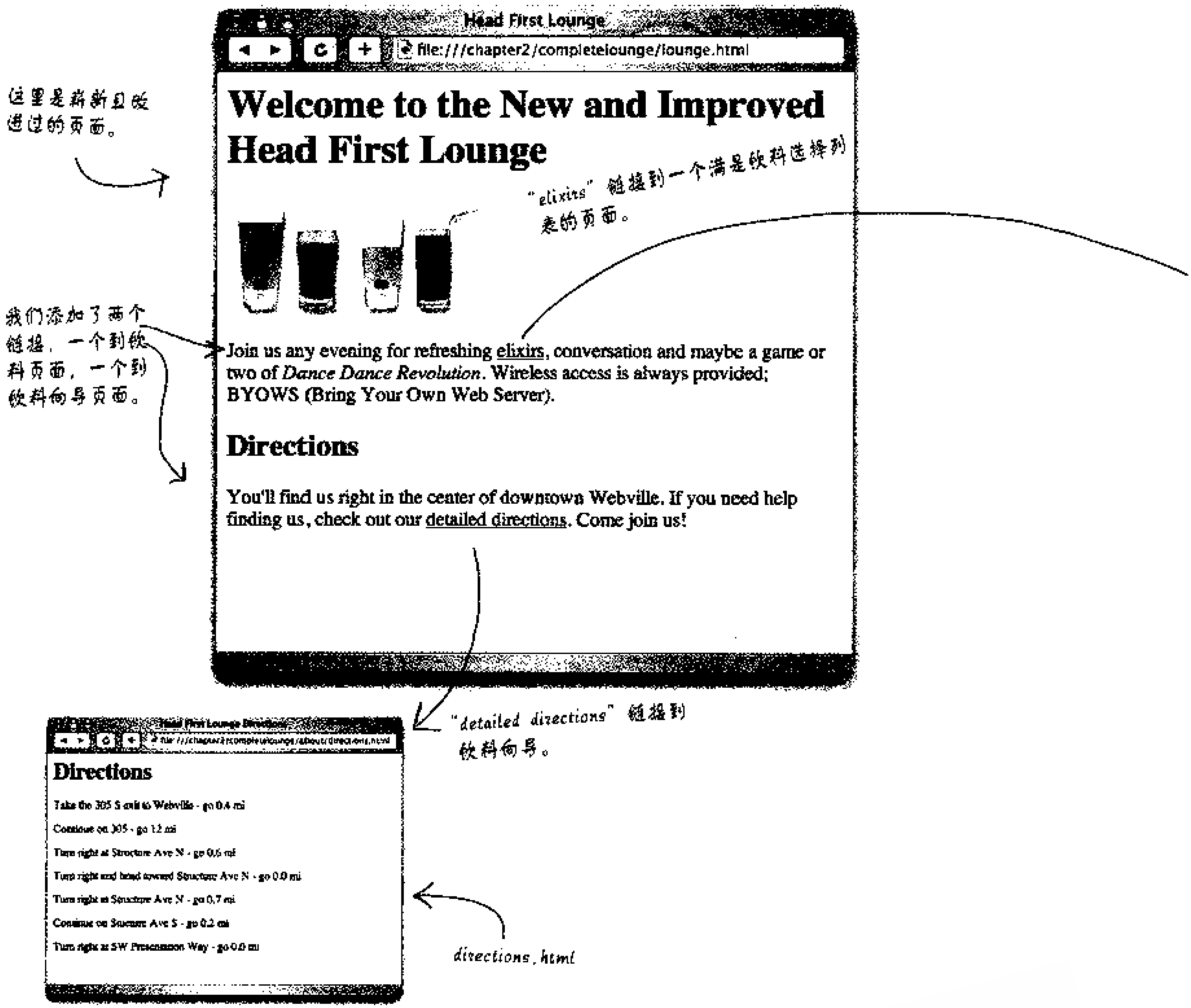
认识HTML中的"HT"

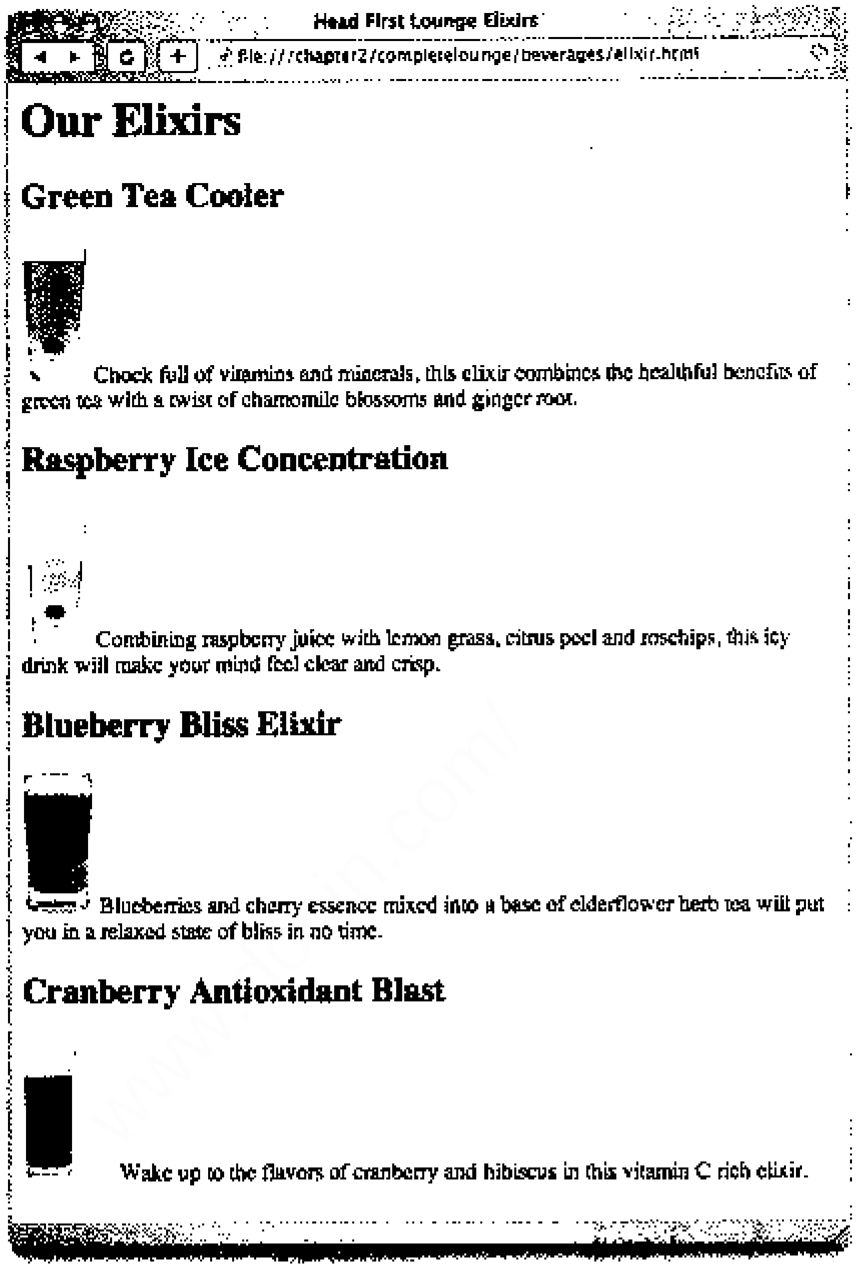


有人提到"hypertext"吗？ 那是什么？哦，它是整个Web的基础。在第1章我们开始学习HTML，并知道了它是一种优秀的描述网页结构的标记语言（即HTML中的'ML'）。现在我们准备学习HTML中的'HT'（即hypertext的缩写），它将引领我们把一个网页链接到其他网页。现在我们学习一个新的元素——<a>元素，并学习"相对"是个多么有用的东西。现在，系紧你的安全带，马上进入超文本的学习。

崭新且改进过的Head First休闲室

记得Head First休闲室么？那可是个好网站，如果顾客可以浏览新推出的饮料列表不是更好吗？甚至，我们可以为顾客提供一些导航，以方便他们找到想要的饮料。





elixir.html

该页介绍新鲜健康的饮料，学习之前，先随便来一杯。

创建崭新且改进过的休闲室需要三步：

让我们再加工原始的Head First休闲室页面，让它链接到两个新的网页。

1. 第一步很简单，因为我们已经为你创建了“directions.html”和“elixir.html”文件，在本书的资源文件中可以找到它们，http://www.headfirstlabs.com。 **待烘烤的**

2. 然后编辑“lounge.html”文件，把“directions.html”和“elixir.html”的链接添加到HTML中。

3. 最后测试网页和链接。完成后，坐下来看看它是如何工作的。

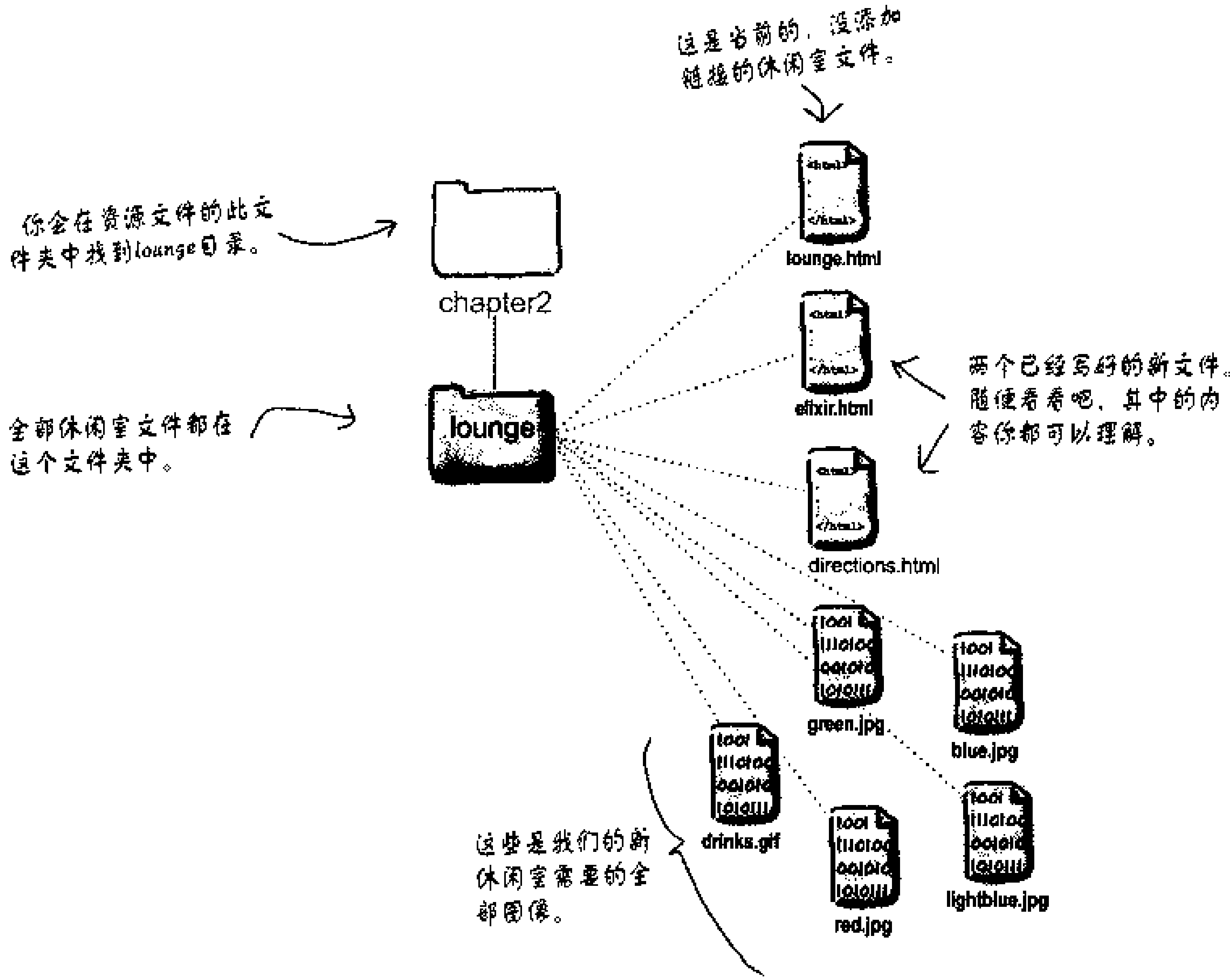
打开网页，开始吧……

创建新的休闲室



❶ 获取资源文件

从以下网站获得资源文件：http://www.headfirstlabs.com。下载完后，你会在“chapter2/lounge”文件夹里找到“lounge.html”、“directions.html”和“elixir.html”这些文件及一些图像文件。



BRAIN POWER

Head First休闲室正在创建中，要很好地组织网站就不能把所有的网站文件放在一个文件夹中。那么，你可以做些什么改进呢？

❷ 编辑"lounge.html"

用编辑器打开"lounge.html"文件，并添加下面突出显示的文本和HTML。完成后，我们将在下一页看看它是如何运行的。

```
<html>
  <head>
    <title>Head First Lounge</title>
  </head>
  <body>
    <h1>Welcome to the New and Improved Head First Lounge</h1>
    <img src="drinks.gif">
    <p>
      Join us any evening for
      refreshing <a href="elixir.html">elixirs</a>,
      conversation and maybe a game or two of
      <em>Dance Dance Revolution</em>.
      Wireless access is always provided;
      BYOWS (Bring your own web server).
    </p>
    <h2>Directions</h2>
    <p>
     You'll find us right in the center of downtown Webville.
     If you need help finding us, check out
     our <a href="directions.html">detailed directions</a>.
     Come join us!
    </p>
  </body>
</html>
```

在标题中添加文本"New and Improved"。

在此添加到饮料的链接。

我们用<a>元素创建链接。随后会学习它的功能。

需要在这里添加些文字以便引导顾客。

在这里再次用<a>元素添加到向导页面的链接。

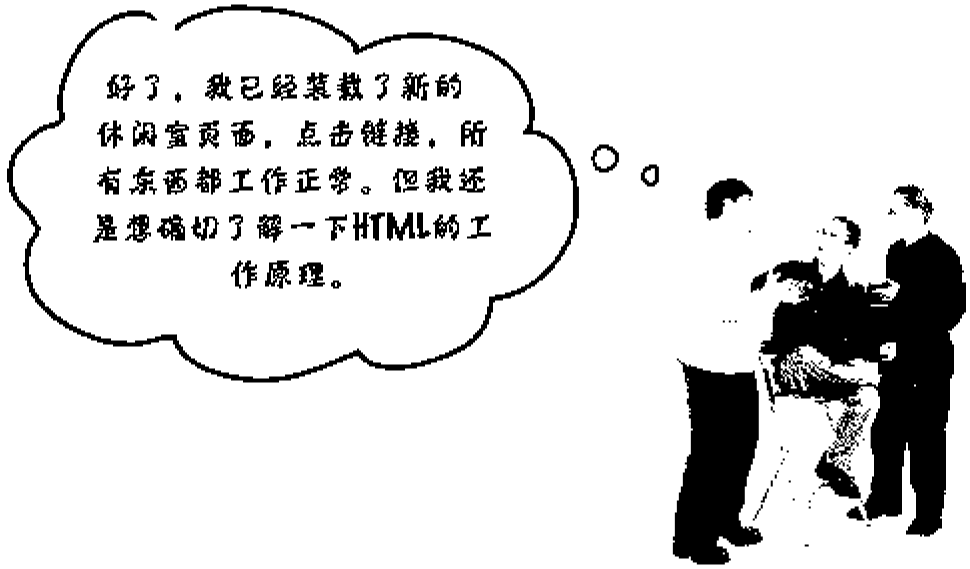
❸ 保存"lounge.html"文件并测试

完成上述更改后，保存并用浏览器打开。下面我们来做些新的尝试……

❶ 点击饮料链接，就会显示新的饮料页面。

❷ 单击浏览器的返回键，会再次显示"lounge.html"。

❸ 点击向导链接，会显示新的向导页面。



幕后花絮

我们做了些什么？

❶ 让我们逐步学习如何创建HTML链接。首先需要把想链接的文本放到<a>元素中，像这样：

```
<a>elixirs</a>          <a>driving directions</a>
```

<a>元素用来创建一个到其他页面的链接。

<a>元素中的内容会作为链接的标题。浏览器将在标题下添加下划线，表示可点击它。

❷ 现在每个链接都有标题了，接着需要添加些HTML来告诉浏览器链接的指向：

在此链接中，浏览器会显示"elixirs"链接，点击将跳转到"elixirs.html"页面。

```
<a href="elixir.html">elixirs</a>
```

Href属性用来指定链接的目的地。

在这个链接中，浏览器将显示"driving directions"链接，点击将跳转到"directions.html"页面。

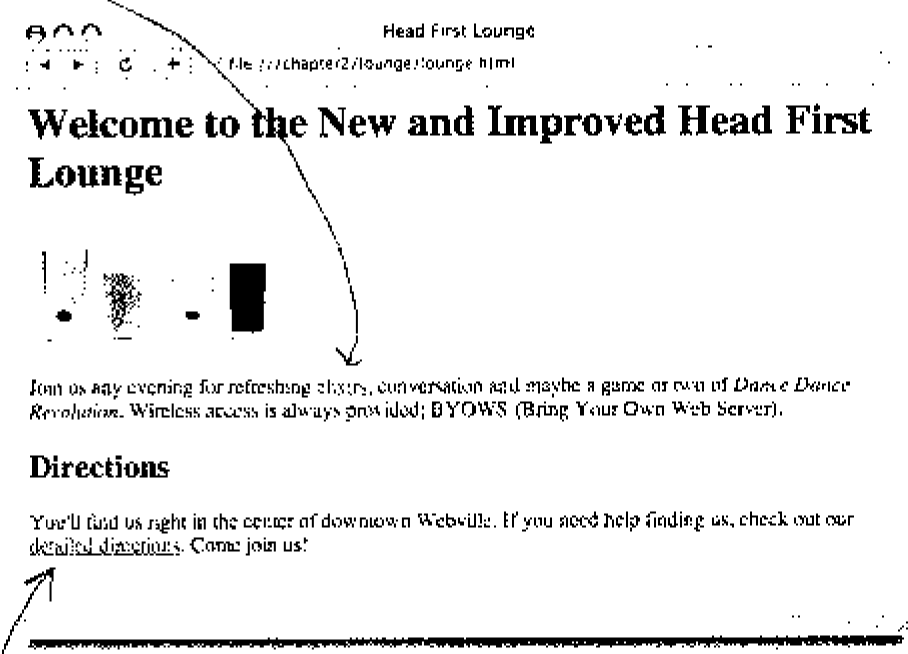
```
<a href="directions.html">driving directions</a>
```

浏览器要做什么？

幕后花絮

1. 首先，浏览器阅读网页，如果网页中包含<a>元素，就把<a>元素中的内容显示为可点击的链接。

```
<a href="elixir.html">elixirs</a>
```



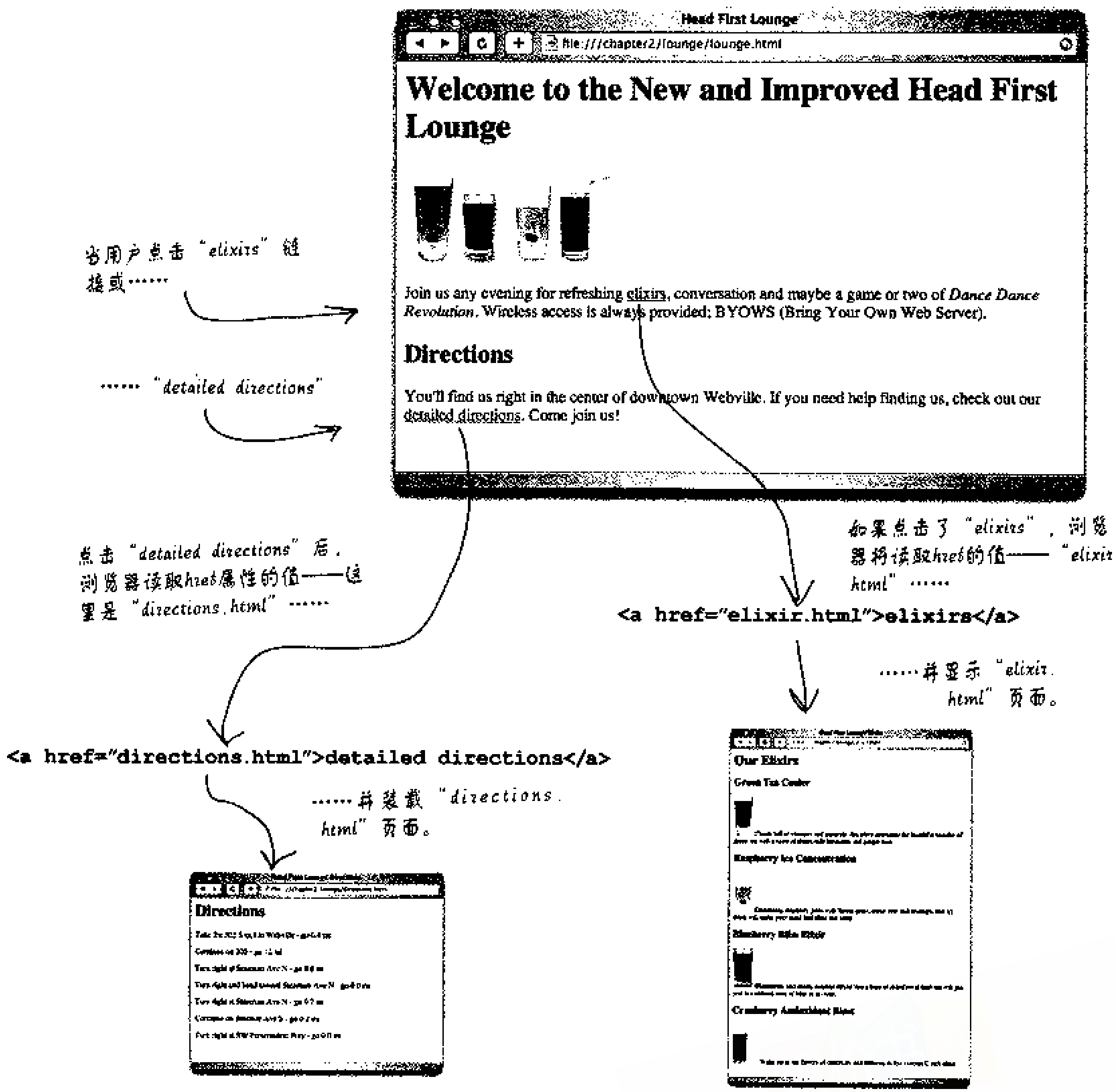
"elixir" 和 "detailed directions" 都包含在<a>开始标记和结束标记中，所以它们在网页中显示为可以点击的链接。

```
<a href="directions.html">detailed directions</a>
```

使用<a>元素创建超文本链接到另外一个网页。
<a>元素中的内容在网页中是可点击的。
Href属性告诉浏览器链接的目的地。

幕后花絮

❷ 然后，用户点击链接，浏览器就使用“href”属性确定链接的目的地。





了解属性

属性用来指定元素的附加信息。虽然我们还没有详细讲解元素，不过你可以先看看有关属性的一些例子：

```
<style type="text/css">
<a href="irule.html">
<img src="sweetphoto.gif">
```

type属性指定使用的样式语言种类，此例中是CSS。

Href属性告诉我们超文本链接的目的地。

src属性指定img标记显示的图片文件名。

我们给出一个例子，让你进一步理解属性的工作方法。

如果<car>是个元素会怎样？

如果<car>是个元素，你会自然地这样标记：

没有任何属性，我们能提供的信息只有车名。

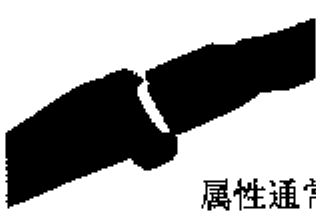
```
<car>My Red Mini</car>
```

但是<car>元素只是提供了一个车名——它没告诉我们想知道的其他信息，如构造、型号、甚至它是敞篷车还是价格昂贵跑车。所以，如果<car>是一个元素的话，我们必须使用以下的属性：

通过使用属性，我们可以用各种信息个性化元素。

```
<car make="BMW" model="Mini Cooper" convertible="no">My Red Mini</car>
```

这样好多了吧？现在标记用一种便于编写、易于组织的形式告诉我们许多有用的信息。



安全第一

属性通常用统一的方法书写：首先是属性名，接下来是等号，然后是用双引号括起来的属性值。

在网上能看到许多结构不严谨的HTML，它们漏掉了许多双引号，你可不要偷懒哦。

结构不严谨会导致你前行时出现许多问题（稍后在书中就会看到）。

这样写（正确的方式）

```
<a href="top10.html">Great Movies</a>
```

属性名　等号　双引号　属性值　双引号

不要这样写（错误的方式）

```
<a href=top10.html>Great Movies</a>
```

错误——属性值周围没有双引号。

there are no Dumb Questions

问： **我能自己创建一个HTML属性吗?**

答： 不行，因为网络浏览器只知道事先指定好的每个元素的属性。如果你创建了属性，浏览器就会不知所措。你在以后章节中会看到，做这类尝试只会徒增烦恼。如果浏览器能认出某元素或者属性，我们就称它支持这个元素或者属性。你只能使用被支持的属性。

问： **谁决定元素是否被支持?**

答： 标准委员会来考虑HTML中的元素和属性是否被支持。这个委员会的成员有大量的时间和精力（没别的事好干），来确保有个HTML准则，使得所有公司都能应用此准则使他们的浏览器工作。

问： **我怎么知道哪个元素或者属性是被支持的? 或者说，任意属性都可以应用到任意的元素上吗?**

答： 只有某些属性才能运用到特定的元素上。这么想：你不会在一个叫<toaster>（烤箱）的元素中用一个叫"covertible（可变的）"的属性吧? 就是说，你只想用有意义并且被元素支持的属性。

我们接下来会学哪些元素对应支持哪些属性，这会贯穿整本书。读完本书后，你还可以借助许多优秀的参考书来巩固记忆，比如《HTML和XHTML权威指南》。





属性揭密

本周访问：
href属性的告白

Head First：欢迎你，href，非常荣幸能采访你这个名声显赫的属性。

href：谢谢。在这里可以摆脱一切链接，感觉真好。跟它们在一起实在是太累了。每次有人点击链接后，我就得告诉浏览器下一步应该干什么。

Head First：感谢你在百忙中挤出时间。让我们回到刚才的问题，作为一个属性，最大的贡献是什么？

href：属性当然是用来个性化元素的啦。用<a>元素随便包围一些内容，如“Sign up now!”——可以这么做：`<a>Sign up now!</a>`——但是如果没有我这个href属性，你就没法告诉浏览器<a>元素的目的地。

Head First：不明白……

href：……你可以通过添加属性来提供额外的元素信息。就我而言，就是链接的指向：

```
<a href="sign up now!">Sign up now!</a>.
```

这表明标记为“sign up now!”的<a>元素链接到sign up now!页面。现在世界上还有许多其他属性，但只有我和<a>元素一起能说明你链接的指向。

Head First：这样啊，那我还得问问，希望您别生气，href这个名字有什么意义吗？

href：这是个因特网家族的旧名字了。意思是“hypertext reference（超文本引用）”，不过我的朋友都简称我为href。

Head First：那又是什么？

href：超文本引用只是你电脑上或因特网上的资源的一个别称，通常是网页，但也可以是音频、视频等。

Head First：听起来挺有趣，我们的读者已经学会了如何链接到自己机器上的网页，但怎么链接到因特网上的网页或者资源呢？

href：嘿，我又得回去工作了。网站没有我就没有意义！对了，教他们如何链接之类的活不是你的工作吗？

Head First：OK，对，我们已经开始教他们入门了，谢谢接受采访。

练习

你已经创建了从“lounge.html”到“elixir.html”和“directions.html”的链接，现在我们开始新的课程。

以下是“elixir.html”的HTML代码，在结尾添加标签为“Back to the lounge（返回休闲室）”的链接，使其指向“lounge.html”。

```
<html>
 <head>
    <title>Head First Lounge Elixirs</title>
  </head>
  <body>
    <h1>Our Elixirs</h1>

    <h2>Green Tea Cooler</h2>
    <p>
            <img src="green.jpg">
            Chock full of vitamins and minerals, this elixir
            combines the healthful benefits of green tea with
            a twist of chamomile blossoms and ginger root.
    </p>
    <h2>Raspberry Ice Concentration</h2>
    <p>
            <img src="lightblue.jpg">
            Combining raspberry juice with lemon grass,
            citrus peel and rosehips, this icy drink
            will make your mind feel clear and crisp.
    </p>
    <h2>Blueberry Bliss Elixir</h2>
    <p>
            <img src="blue.jpg">
            Blueberries and cherry essence mixed into a base
            of elderflower herb tea will put you in a relaxed
            state of bliss in no time.
    </p>
    <h2>Cranberry Antioxidant Blast</h2>
    <p>
            <img src="red.jpg">
            Wake up to the flavors of cranberry and hibiscus
            in this vitamin C rich elixir.
    </p>


  </body>
</html>
```

在这里写你的新HTML。

完成后，对“directions.html”做同样的工作。

CONSTRUCTION ZONE BEGINS

我们需要在你的帮助下构造和解构<a>元素。希望新学的关于<a>元素的知识对你有帮助。以下的每行中有一些标签、目的地，还有完整的<a>元素的组合。请填入空缺的信息，第一行已完成。

标签	目的地	你在HTML中要写的
Hot or Not?	hot.html	Hot or Not?
Resume	cv.html	
	candy.html	Eye Candy
See my mini	mini-cooper.html	
let's play		__________

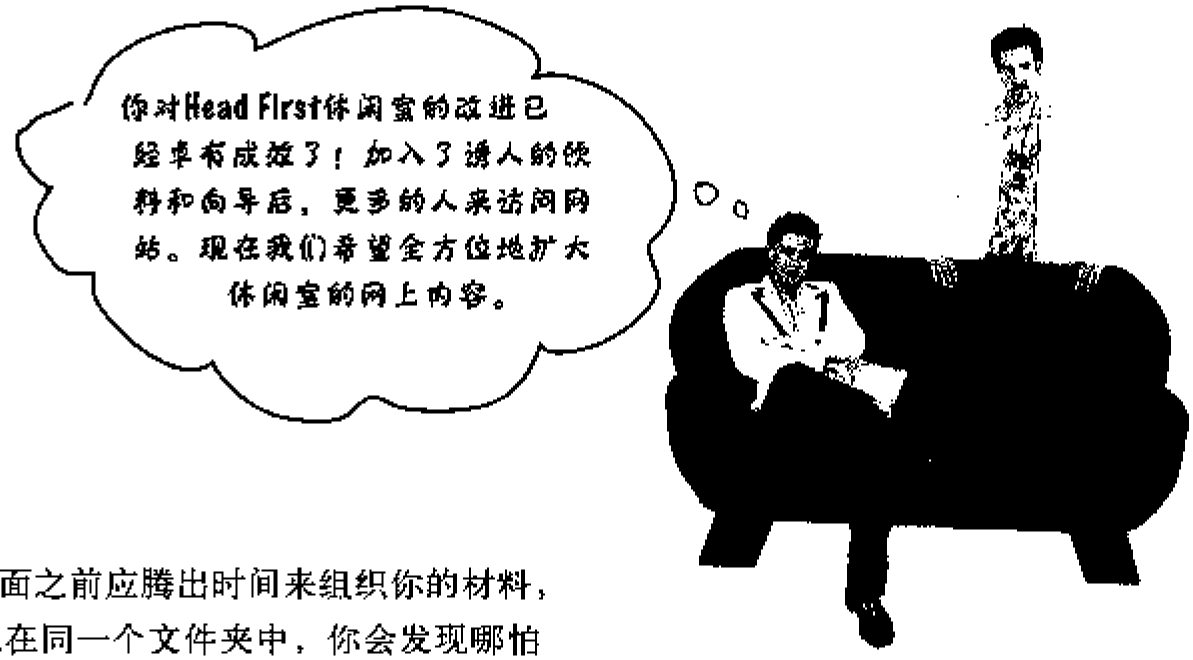
there are no Dumb Questions

问： 我看见许多网页上的图像也是可点击的，我也能用<a>元素来实现吗？

答： 对，如果把<img>元素放进<a>标记之间，图像就能像文字一样可点击。在以后的章节中会深入学习关于图像的内容，图像的效果和链接一样好。

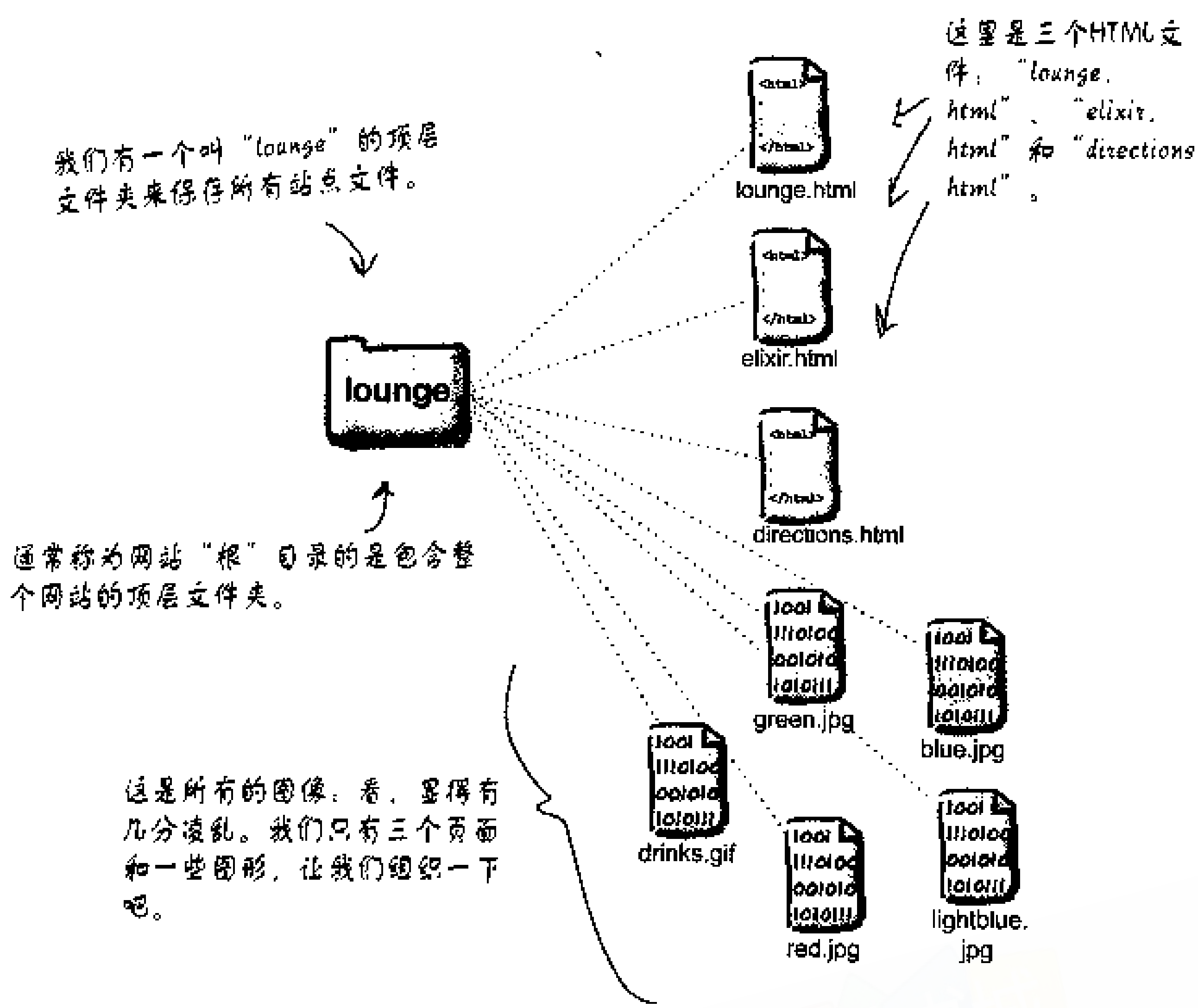
问： 可以在<a>标记之间加入任意内容使其变得可点击吗？比如说，一个段落？

答： 现在先别急。不是任何内容都可以放到<a>元素中的。通常只能是文字和图像（或者两者都有），在标签中嵌入标签完全是另外一个话题，我们很快就会学到的。



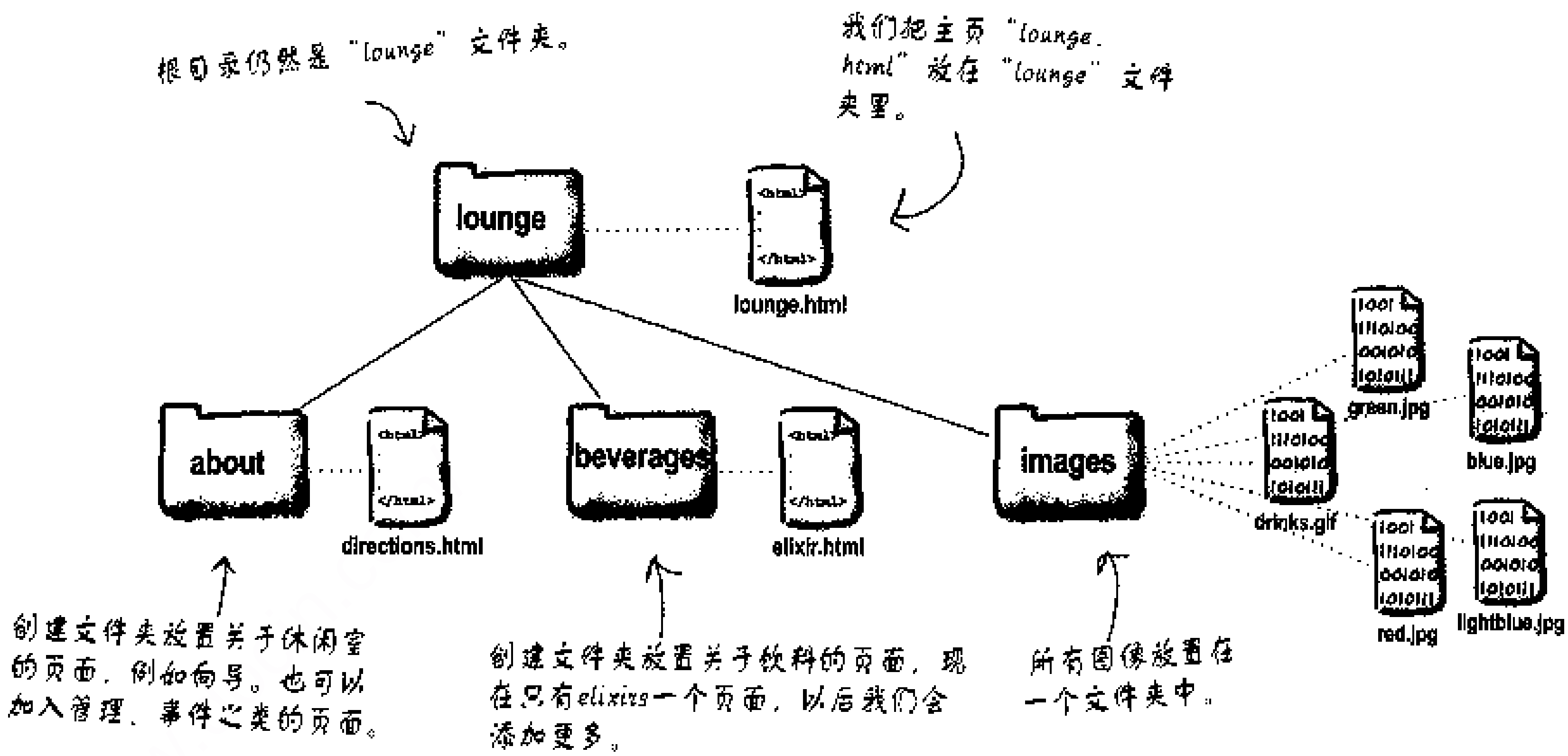
组织

在开始创建更多HTML页面之前应腾出时间来组织你的材料，现在所有的文件和图像还在同一个文件夹中，你会发现哪怕是最小的网站，网页、图像和其他资源分类放置会令你的工作免除后顾之忧。以下是我们现有的：



组织休闲室……

现在让我们对休闲室做些有意义的组织工作。组织网站的方法有很多，我们采用比较简单的。创建一些文件夹来分别放置网页，并把所有图像放到一个文件夹中。



there are no Dumb Questions

问： 既然图像都放在一个文件夹中，为什么不创建一个名为HTML的文件夹来放置所有的HTML文件呢？

答： 可以，组织网站没有所谓绝对正确的方式，更多的是考虑如何方便你和浏览者工作。跟大多数设计者一样，你也希望能尽量保持简单并且灵活的组织以便网站扩建。

问： 或者说，为什么不在每个文件夹下加个图像文件夹，例如在"about"和"beverages"下？

答： 让我们再次肯定地说，可以。有时我们希望某些图像能在几个网页中重复使用，所以把图像全部放到根目录（最顶层）下的一个文件夹里。如果你的网站需要在不同的部分使用大量的图像，就可以在每个枝叶目录下设置各自的图像文件夹。

问： 枝叶？

答： 描述文件夹的方式就像一棵倒栽的树，顶部为根，到下边文件或目录的每条路径就是枝叶。



练习

现在要为前边的网页创建文件和文件夹。以下是你要做的：

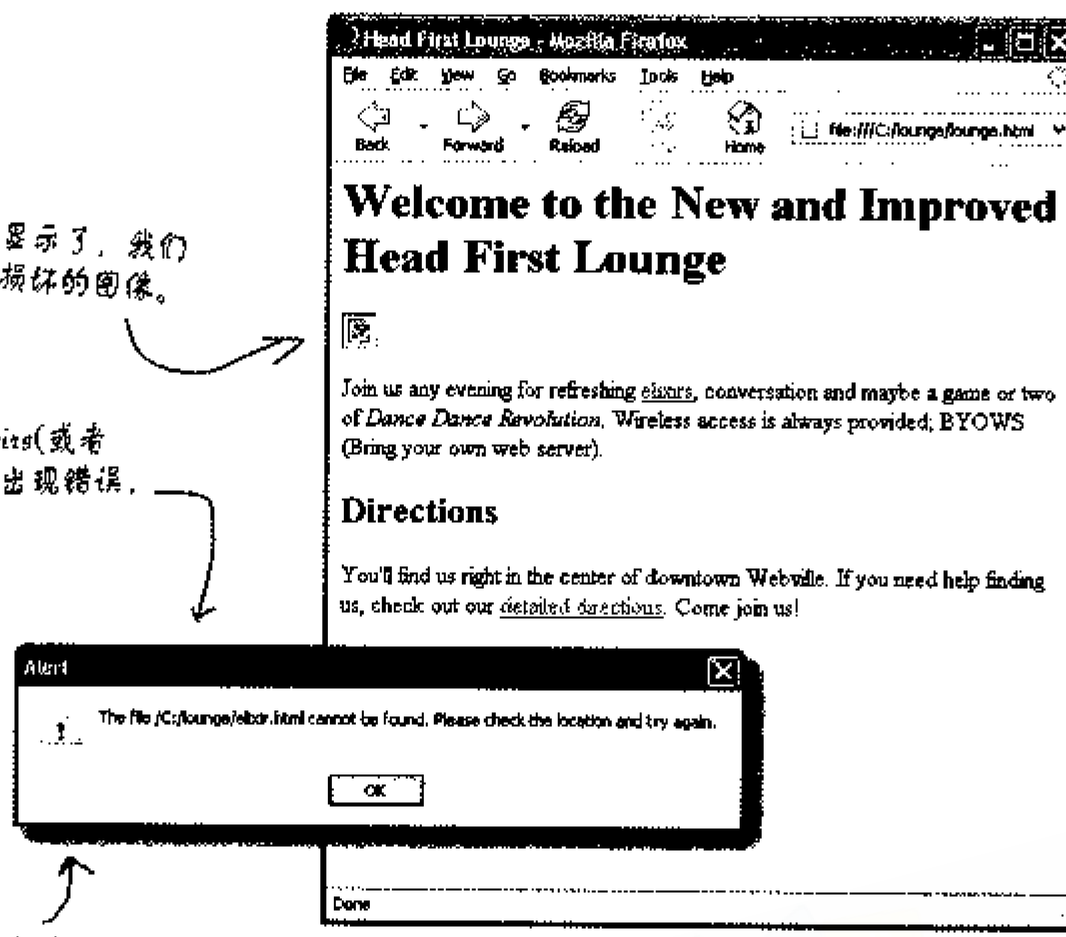
1. 定位lounge文件夹，然后创建新的子文件夹并命名为“about”、“beverages”和“images”。
2. 移动文件“lounge.html”到“about”文件夹。
3. 移动文件“elixir.html”到“beverages”文件夹。
4. 移动所有图像到“image”文件夹。
5. 最后，载入“lounge.htm”文件并实验链接，和下边的答案比较一下。

技术难点：

改动后网页似乎出现了些问题。

有个图像不显示了，我们通常称其为损坏的图像。

而且当我们点击链接elixirs(或者directions)时，更糟糕，出现错误，说找不到该网页。



某些浏览器会用网页而不是对话框来报错。



对，我们需要告诉浏览器页面的新位置。

到现在为止你已学会使用href值来指向同一文件夹下的页面，网站通常会比较复杂，所以你需要能指向其他文件夹下的页面。

为此，你必须把路径从页面指向目的文件，这就是说可以上溯或者下探一个或两个文件夹，但不管用哪种方法我们都必须把相对路径放到href中去。

www.docin.com

计划好你的路径……

当假期和家人开车出游，有哪些事情需要计划？准备地图，并从你的所在地出发，经过一段路程后到达目的地，方向由你的所在地决定，如果你在另外一个城市，方向就改变了，对不对？

OK，你已经准备好了Google这张地图，但是你还是要跟好我们哦。



指出链接的相对路径，跟外出旅游是一回事，从有链接的页面开始，在文件夹里经过一段路径，直至找到所需文件。

还有一种路径，以后的章节中会讲到。

让我们解决一些相对路径（同时完善休闲室）吧。

链接到子目录里

① 从lounge.html链接到elixir.html。

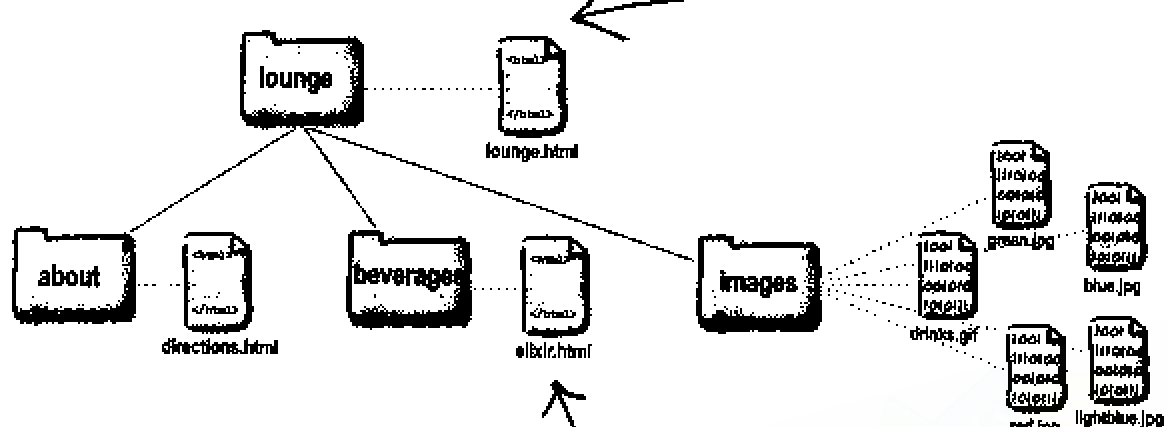
我们需要修正lounge.html页面中的饮料链接，<a>元素现在是这样写的：

```
<a  href="elixir.html">elixirs</a>
```

现在只用文件名“elixir.html”来告诉浏览器它和“lounge.html”在同一个文件夹下。

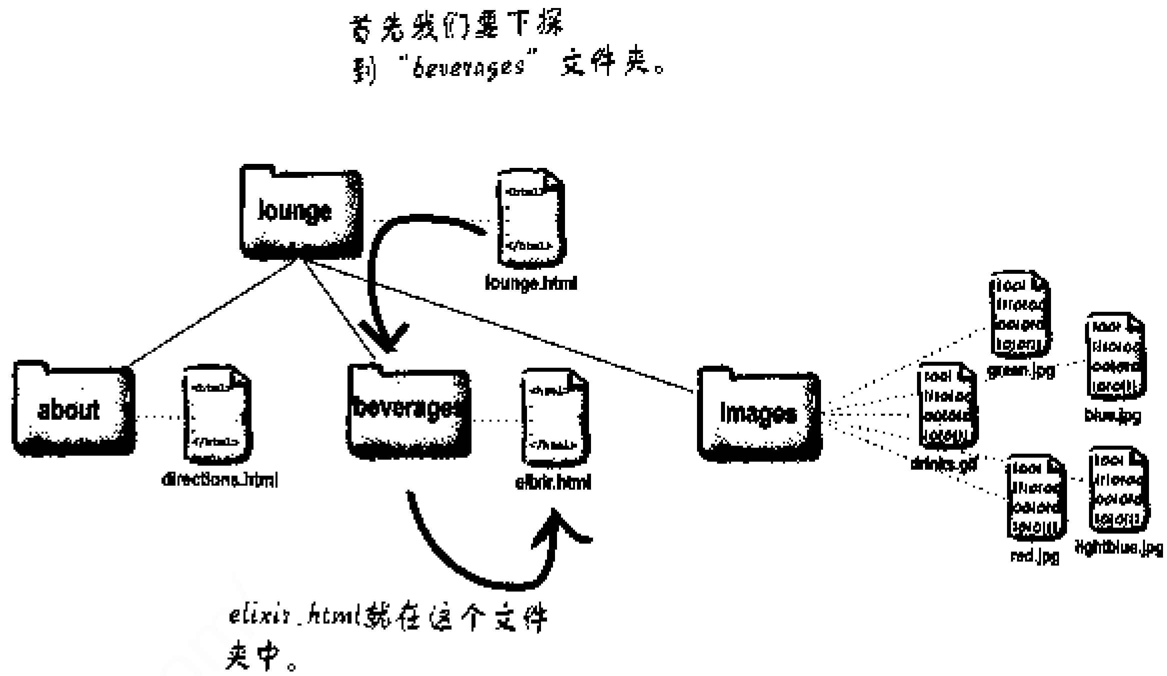
② 确定源文件和目的文件。

当我们重组休闲室后，把“lounge.html”放在“lounge”文件夹中，把“elixir.html”放在“lounge”下的“beverages”文件夹中。



③ 追踪从源文件到目的文件的路径。

我们来追踪路径。要从“lounge.html”到“elixir.html”，我们需要先下探到“beverages”文件夹，然后找到其中的“elixir.html”文件。



④ 创建一个href来显示我们经过的路径。

明白路径后，我们需要把它写成浏览器能理解的格式。这样描述路径：

首先进入beverages文件夹。

用“/”分隔路径的各个部分。

最后写文件名。

```
beverages / elixir.html
```

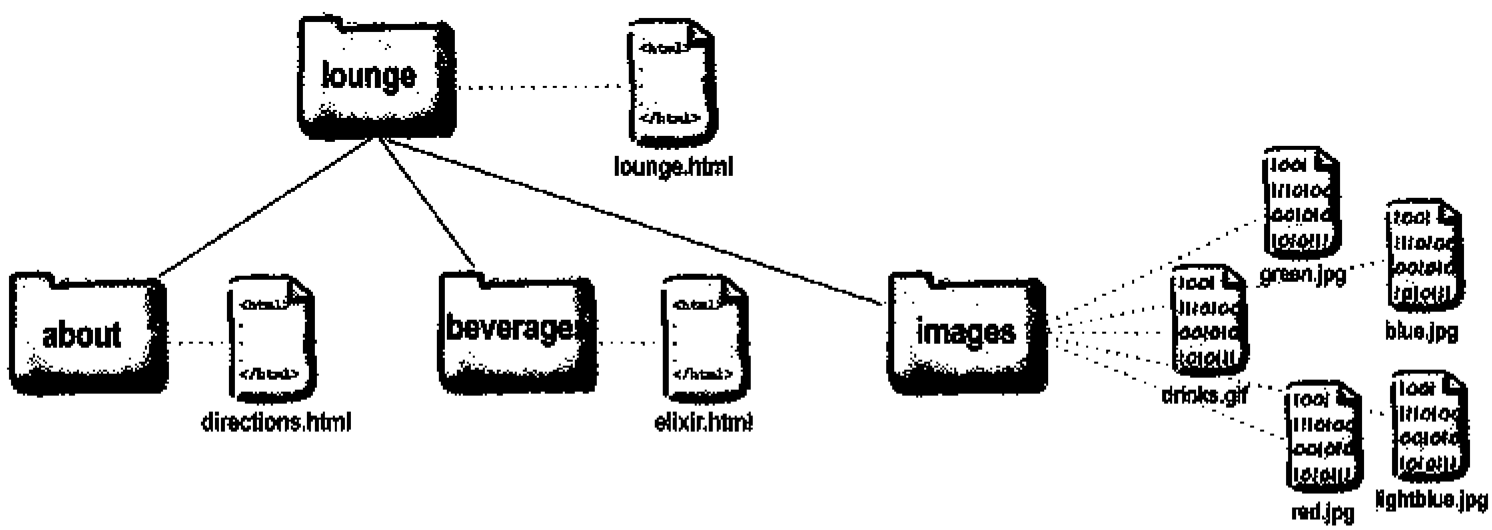
把这些放一起……

```
<a  href="beverages/elixir.html">elixirs</a>
```

我们把相对路径放到href值中，现在如果你点击链接，浏览器将在“beverage”文件夹中寻找“elixir.html”。

Sharpen your pencil

轮到你了：追踪从“lounge.html”到“directiongs.html”的相对路径，找到后，完成以下的<a>元素。参考本章后的答案，然后更改“lounge.html”中的<a>元素。




```
<a  href="______________________">detailed directions</a>
```

把答案写在这里

用另外一种方法：链接到“父”文件夹

① 从“directions”链接到“lounge.html”。

现在我们需要修正“back to the lounge”（返回休闲室）的链接，这里是“directions.html”文件中<a>元素的写法：

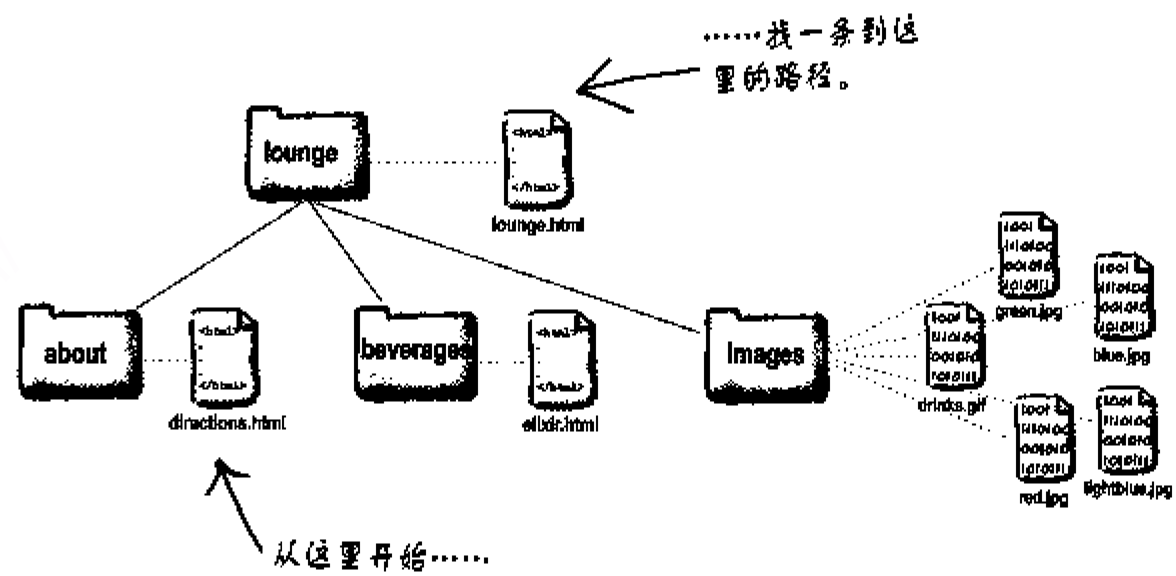
```
<a  href="lounge.html">Back to the Lounge</a>
```

现在我们告诉浏览器在“lounge.html”所在的文件夹下寻找“directions.html”文件。但是这个已经失效了。

② 确定源文件和目标文件。

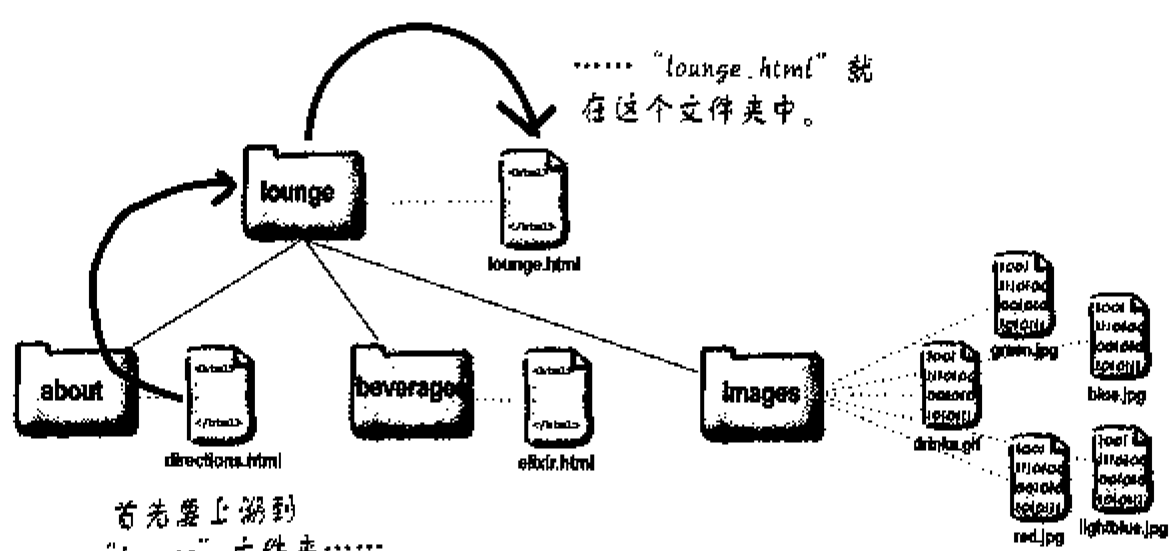
我们来看看源文件和目标文件。源文件现在是“directions.html”，在“about”文件夹中。

目标文件是“lounge.html”，在“about”文件夹的上层文件夹中。



③ 追踪从源文件到目标文件的路径。

我们来追踪路径。要从“directions.html”文件到“lounge.html”，我们需上溯一个文件夹到“lounge”文件夹，然后在该文件夹中寻找“lounge.html”文件。



④ 创建一个href来显示我们经过的路径。

我们快要完成了。知道路径后，把它变成浏览器可以理解的格式，我们来书写路径：

首先，上溯一个文件夹。怎么上溯？用".."，对，两个英文句号。先这么写吧，稍后我们会解释。

用"/"符号分隔路径各部分。

最后写文件名。

```
..  / lounge.html
```

".."读为"点点"。

把这些放在一起……

```
<a href="../lounge.html">Back to the Lounge</a>
```

单击链接后，浏览器便在上一级文件目录中搜索"lounge.html"。



there are no Dumb Questions

问： **什么是父目录？如果我的“fruit”文件夹下有个“apple”文件夹,那么“fruit”就是“apple”的父亲？**

答： 对，我们经常用家庭成员来描述文件夹之间的关系（也许你听过目录）。举例来说，就如你刚刚问的，“fruit”是“apple”的父亲，“apple”是“fruit”的孩子，如果还有别的子文件夹比如“pear”也是“fruit”的孩子，那它就是“apple”的兄弟，像家谱一样。

问： **OK，我明白父目录的意思了。那什么是“..”？**

答： 当你需要告诉浏览器链接的文件在父目录时，使用“..”表示“上溯到父目录”，这就是浏览器所谓的父亲。

在例子中，我们希望从“about”目录下的“directions.html”链接到“lounge”目录下（即“about”的父目录）的lounge.html。因此我们必须告诉浏览器上溯一个文件夹。用“..”来告诉浏览器上溯的方式。

问： **你如果需要上溯的文件夹不只一个而是有两个时，该怎么办？**

答： 可以每上溯一层使用一次“..”。每用一次“..”就上溯了一层父目录。所以，当你想上溯两层目录时，可以这么写“../..”，还是要用“/”来分隔各个部分，不要忘记了（浏览器不能识别“…”）。

问： **我上溯了两层目录，怎么告诉浏览器从哪找文件啊？**

答： “../..”和文件名要结合使用。如果你要链接到一个需要上溯两层的名为“fruit.html”的文件，要写成“../../fruit.html”的形式。你也许认为我们会称“../..”为“祖父”目录，但是通常，我们称其为“父目录的父亲”，或者简称为“../..”。

问： **上溯有没有最大上限啊？**

答： 可以上溯直至你网站的根目录。在我们的例子中，根目录就是“lounge”文件夹。所以，你最多可以上溯到“lounge”文件夹。

问： **另外方面，下探的限度是多少？**

答： 嗯，下探的深度由你创建的文件夹的多少来决定。如果你创建的目录有10层深，那你的路径可以写10层目录。不过我们不推荐这么做——当有太多级目录文件夹时，可能代表你的网站组织过于复杂了。

另外，路径还有字数的限制，最多255个。这是个很大的数字，所以一般不会全用完，但是如果你的网站十分庞大，就要注意了。

问： **我的操作系统用“\”做分隔符，我能用它来代替“/”吗？**

答： 不能。在网页设计中只能使用“/”，而不能用“\”。许多操作系统使用不同的文件分隔符（例如Windows用“\”代替“/”）。但是在网站上，我们挑一个普通的分隔符，大多数人都坚持使用它。所以，无论你使用Mac、Windows、Linux还是其他系统，在你的HTML路径中必须使用“/”分隔符。

BRAIN POWER

轮到你了：写出从“elixir.html”到“lounge.html”的“Back to the Lounge”的相对路径。看看和在“directions.html”中的链接有什么不同？

答案：没有，两个路径几乎是一样的。

修复损坏的图像……



休闲室已经创建的差不多了，现在你要做的是修复那些不能正常显示的图像。我们还没有细学**<img>**元素（过几章就会学到），但我们要知道**<img>**元素的**src**属性也有和**href**属性类似的相对路径。

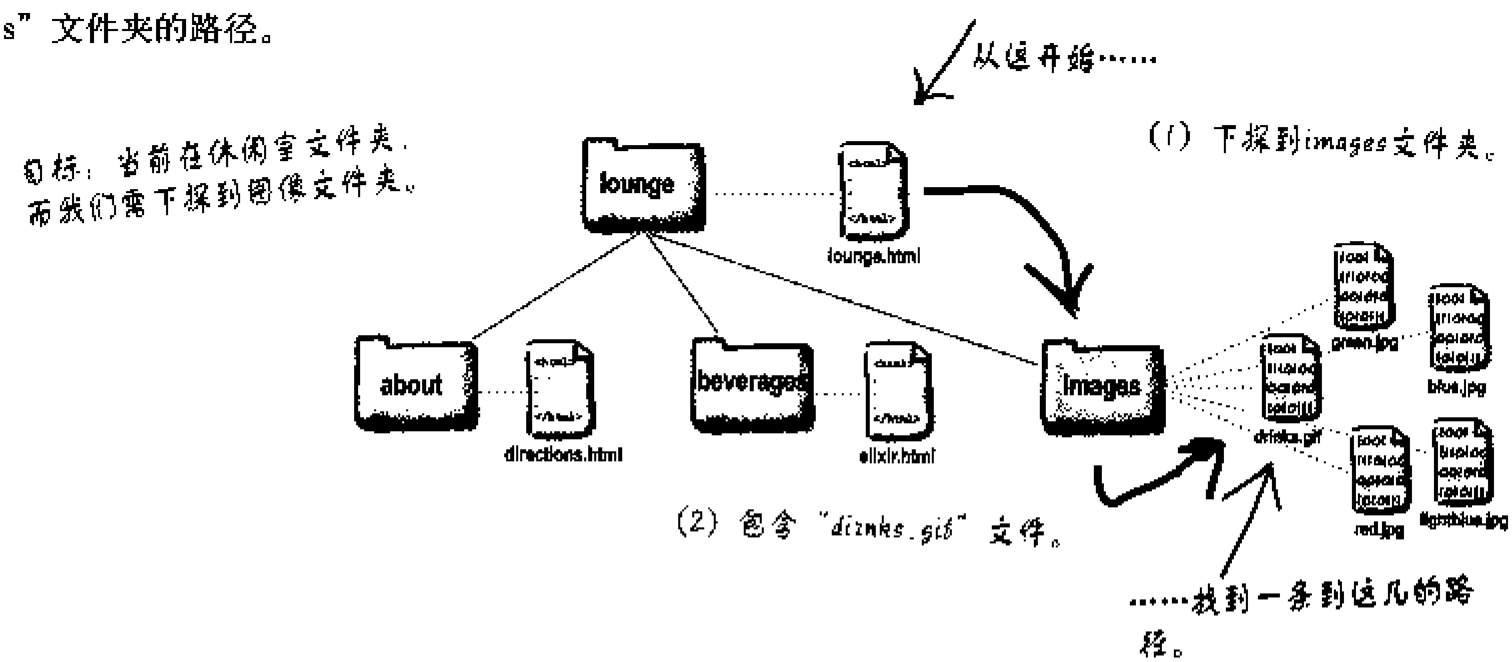
这是“lounge.html”中的图像元素：

```
<img src="drinks.gif">
```

这里告诉浏览器图像位置的相对路径。我们要像<a>元素中的href属性那样重新指定相对位置。

找到从“lounge.html”到“dirnks.gif”的路径

我们需要找到从“lounge.html”到图像所在的位置，即“images”文件夹的路径。

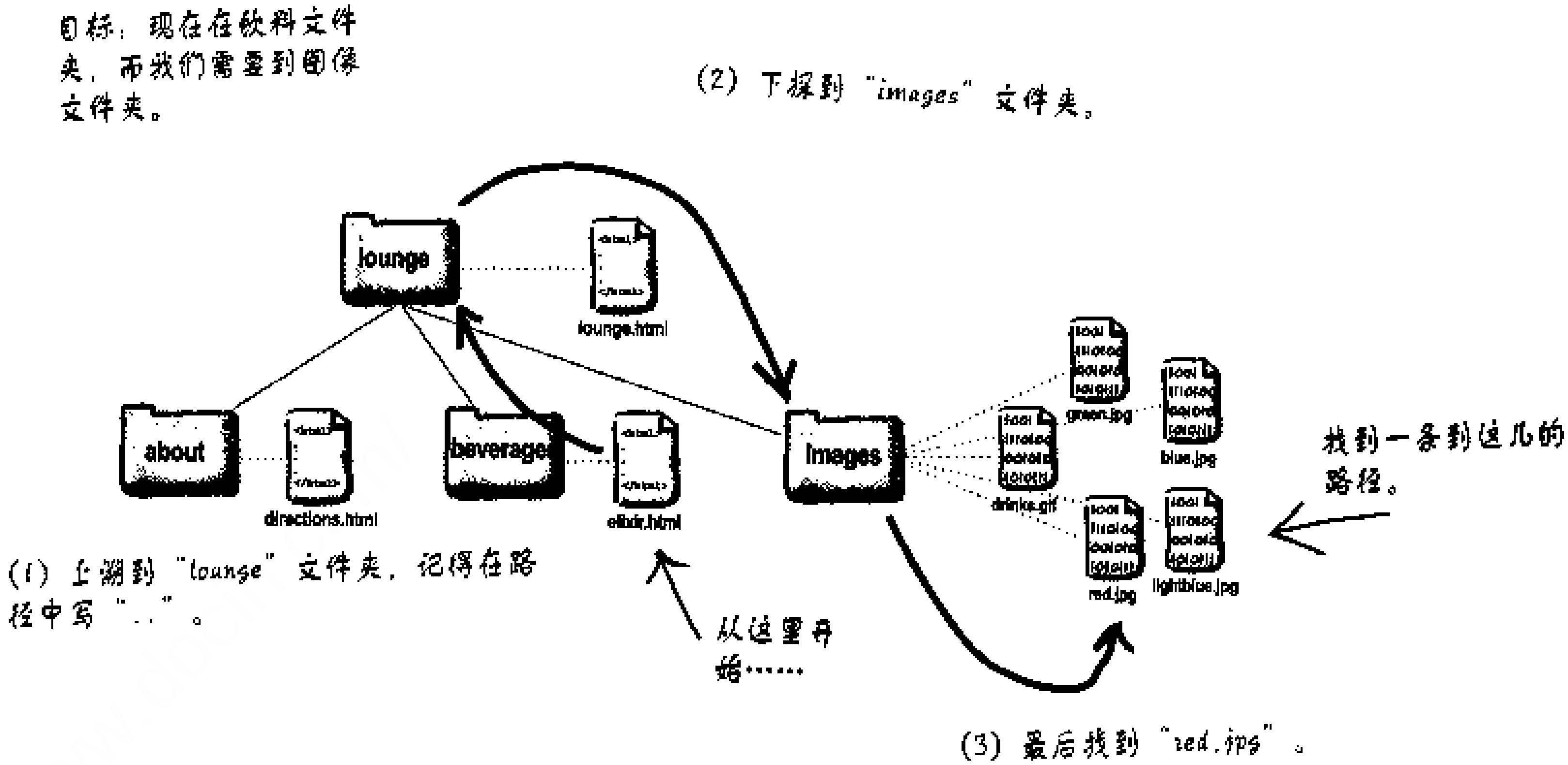


所以把（1）和（2）合在一起就是我们要找的路径，例如“images/drinks.gif”或者：

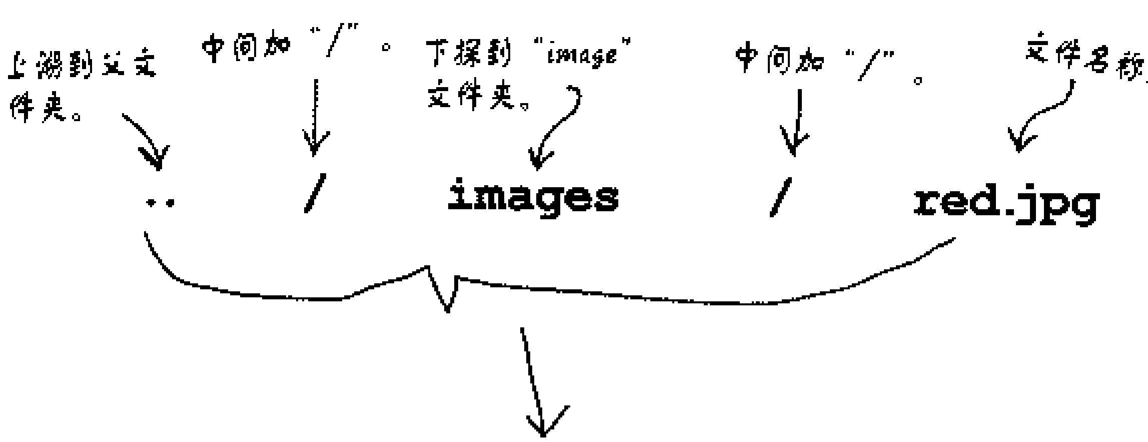
```
<img src="images/drinks.gif">
```

找到从"elixir.html"到"red.jpg"的路径

Elixirs页面包含几种饮料的图像，red.jpg、green.jpg、blue.jpg之类。我们來指出到red.jpg的路径，其他图像的路径是相似的，因为它们都在同一个文件夹里。



将（1），（2），（3）合一起我们得到：

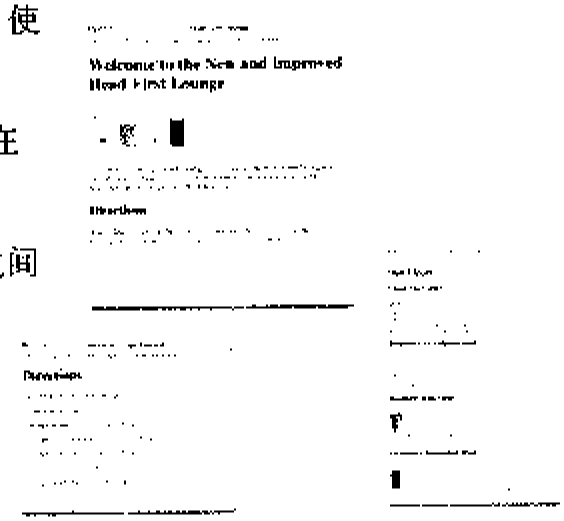



```
<img src="../images/red.jpg">
```

练习

现在已经修复了重新组织休闲室时损坏的所有链接，然而，我们还要修复“lounge.html”和“elixir.html”文件中的图像。以下是你要做的：

1. 在“lounge.html”中更改图像的src属性，使其值为“images/drinks.gif”。
2. 在“elixir.html”中更改图像的src属性，在每个图像文件名前加“../images/”。
3. 保存，载入浏览器，现在你能在各页面之间自由跳转并观看图像了。



附言：如果还有什么困难的话，在文件夹“chapter2/completelounge”中可以找到完成的休闲室文件。进行对比再次检查你的成果。



要点

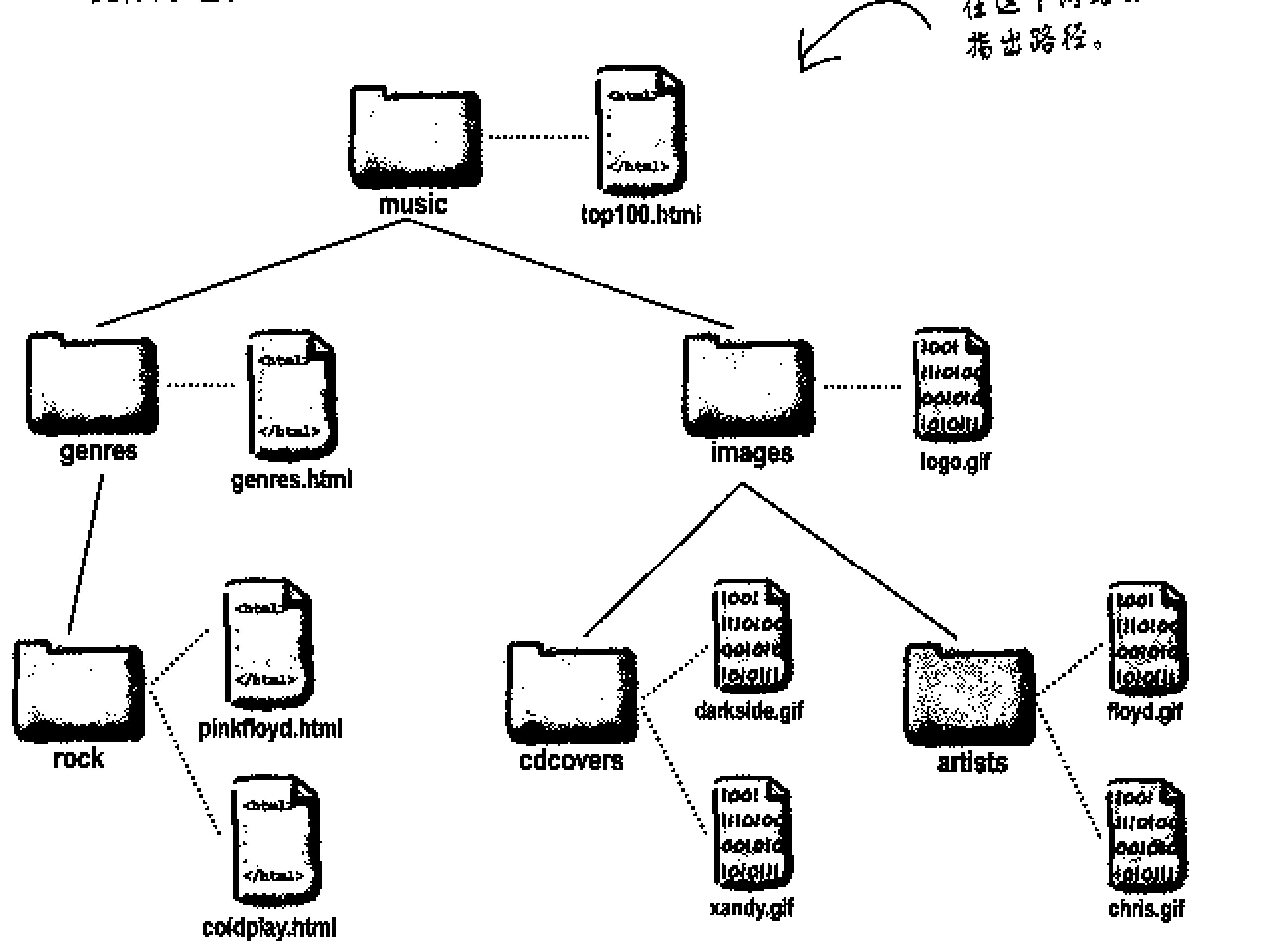
- 当你需要从一个网页链接到另一个网页时，使用<a>元素。
- <a>元素中的href属性说明了链接的目的地。
- <a>元素中的内容是链接的标签。标签显示在网页上。默认情况下，它会加下划线来说明是可以单击的。
- 链接的标签可以是文字或图像。
- 当你点击链接时，浏览器装载href中指定的网页。
- 可以链接相同文件夹下的文件，也可以是不同文件夹下的。
- 相对路径是指向网站中的其他文件的链接，跟链接的源文件有关。就像地图一样，与终点和起点有关。
- 使用“..”来链接父目录中的文件。
- “..”代表父目录。
- 记得用“/”符号分隔路径的各个部分。
- 当图片的路径有误时，你会在网页上看到损坏的图像。
- 网站中的文件名或者文件夹名中不能出现空格。
- 最好在构建的早期组织好你的网站，这样当网站升级时就不用更改一堆路径了。
- 组织网站的方法有许多，怎么做取决于你。

相对性的重大挑战

现在有一个机会来测试你的相对性技能：我们在一个叫“music”的文件夹下放了个关于100套唱片的网站。在这个文件夹下有HTML文件、其他文件夹和图像。你的任务是找到我们需要的相对路径，以便我们能从网页链接到其他网页和文件。

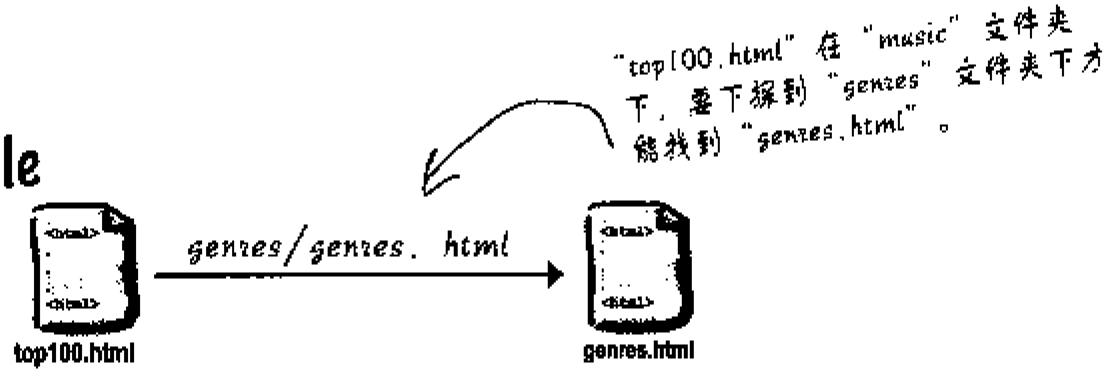
在这一页，你会看到网页的结构，在下一页你会看到测试你能力的题目。你的工作是更正每个源文件和目标文件的相对路径。如果成功了，你将成为相对路径测试的冠军。

祝你好运！

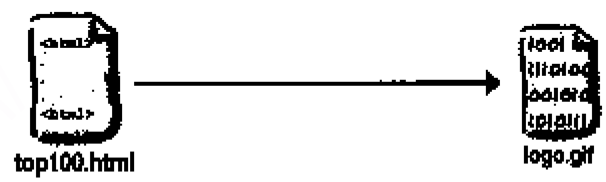


竞赛该开始了。准备……开始……写！

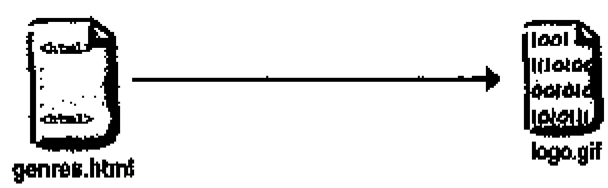
Example



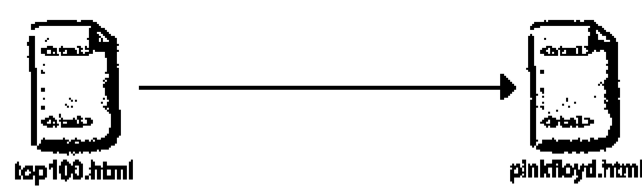
Round One



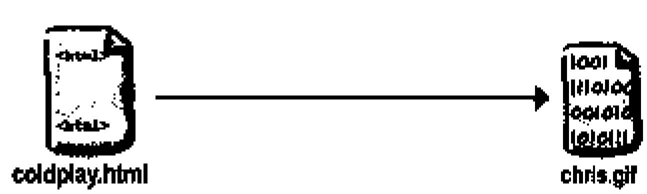
Round Two



Round Three



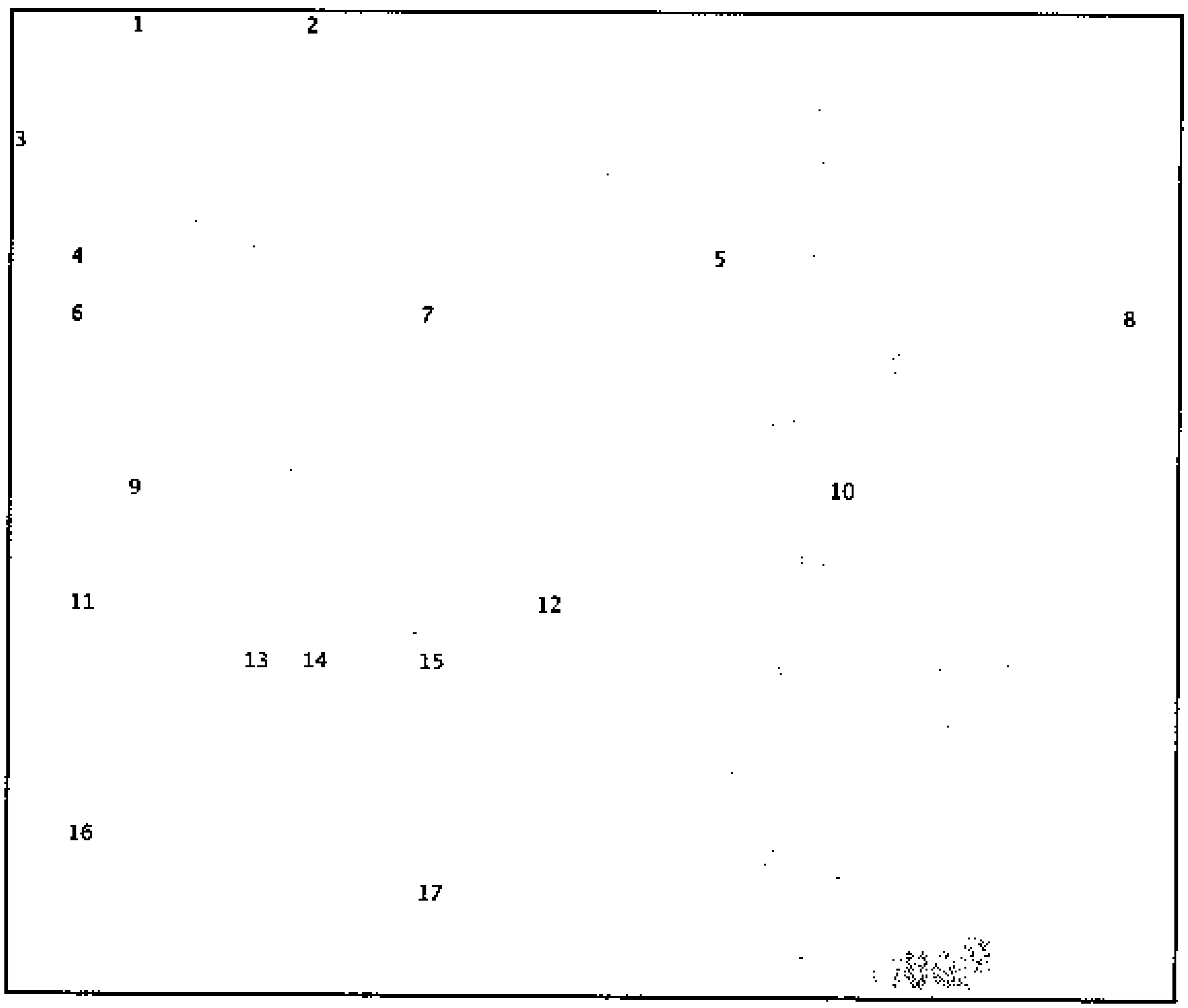
Bonus Round





HTML填字游戏

纵横字谜是如何帮助你学习HTML的？所有的词都与这章的HTML有关。另外，对线索的思考能帮助你开拓大脑右半球。



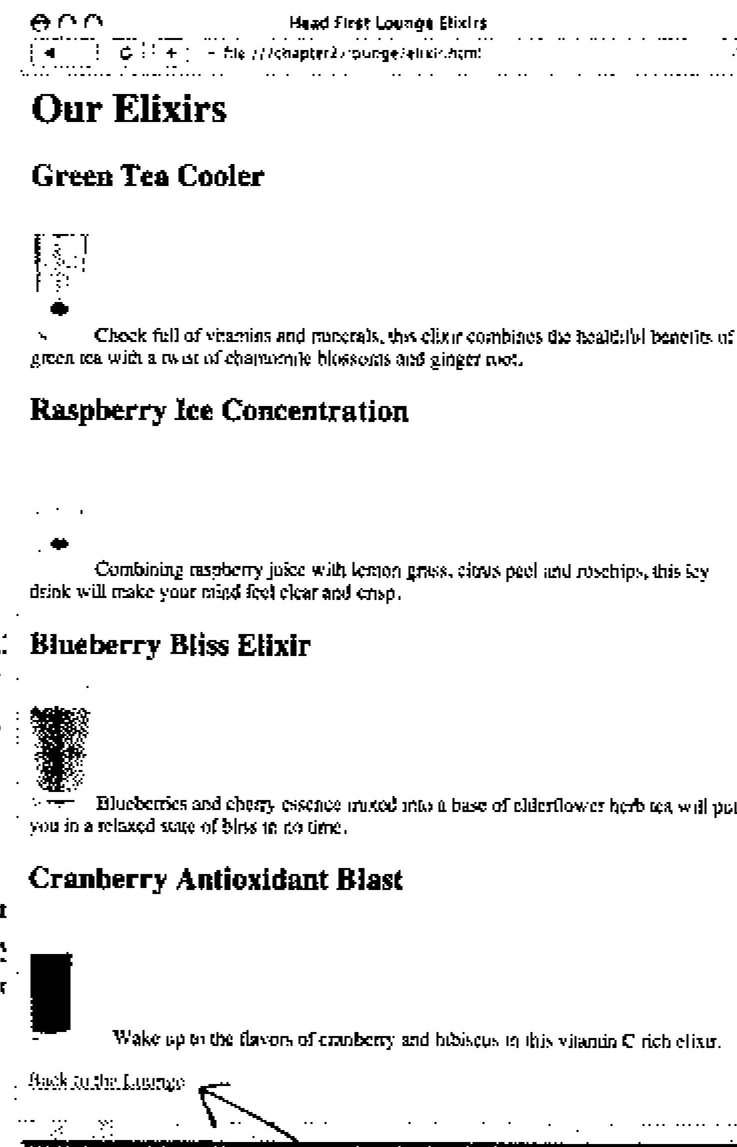
横排提示：

1. ../myfiles/index.html是这种形式的链接。
3. 文件夹的另外一个名称。
6. blue drink的味道。
9. href代表的意义。
13. 在<a>和之间的内容。
16. 像文字一样可以加在<a>中的。
17. “..”的发音。

竖排提示：

2. href和src是其中的两种。
4. 网站中最努力的工作属性。
5. href的韵律。
7. 网站最顶层的目录。
8. HTML中的“H”。
10. 健康饮品。
11. 同级的文件夹。
12. 使用..到达的地方。
14. 在<a>标签中的文本作为______。
15. 子目录也叫这个名字。

练习答案




```
<html>
  <head>
    <title>Head First Lounge Elixirs</title>
  </head>
  <body>
    <h1>Our Elixirs</h1>

    <h2>Green Tea Cooler</h2>
    <p>
            <img src="green.jpg">
            Chock full of vitamins and minera
            combines the healthful benefits of
            a twist of chamomile blossoms and
    </p>
    <h2>Raspberry Ice Concentration</h2>
    <p>
            <img src="lightblue.jpg">
            Combining raspberry juice with le
            citrus peel and rosehips, this ic
            will make your mind feel clear an
    </p>
    <h2>Blueberry Bliss Elixir</h2>
    <p>
            <img src="blue.jpg">
            Blueberries and cherry essence mixed into a base
            of elderflower herb tea will put you in a relaxed
            state of bliss in no time.
    </p>
    <h2>Cranberry Antioxidant Blast</h2>
    <p>
            <img src="red.jpg">
            Wake up to the flavors of cranberry and hibiscus
            in this vitamin C rich elixir.
    </p>
    <p>
      <a href="lounge.html">Back to the Lounge</a>
    </p>
  </body>
</html>
```

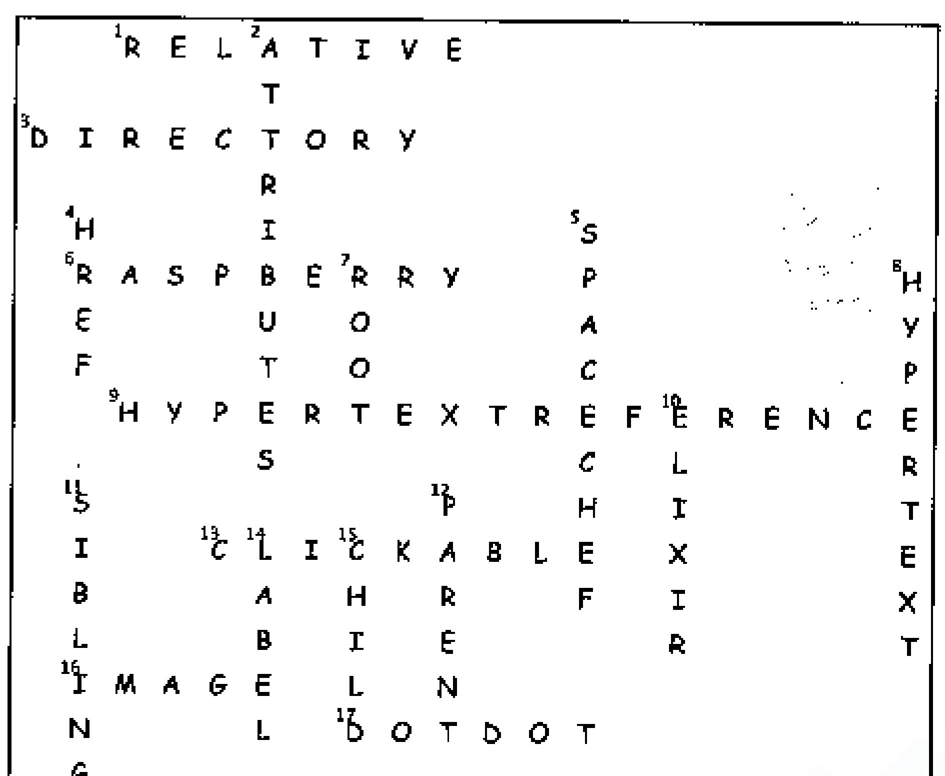
这里是新的<a>元素，指回休闲室。

我们把链接放在单独的一段，这样显得整齐些。我们会在下一章谈更多这类问题。

习题解答

CONSTRUCTION ZONE BEGINS

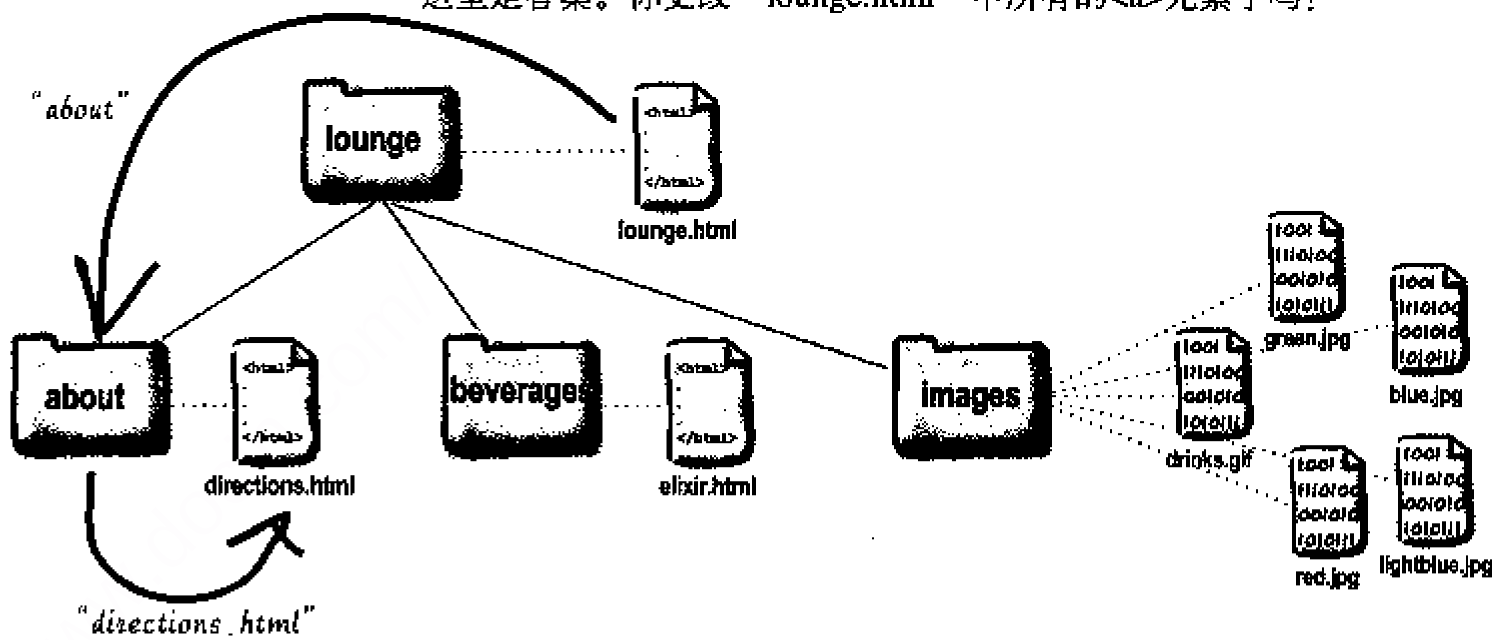
标签	目的地	元素
Hot or Not?	hot.html	Hot or Not?
Resume	cv.html	Resume
Eye Candy	candy.html	Eye Candy
See my mini	mini-cooper.html	See my mini
let's play	millionaire. html	 let' s play



Sharpen your pencil 答案

追踪从“lounge.html”到“directiongs.html”的相对路径，找到后，完成下面的<a>元素。

这里是答案。你更改“lounge.html”中所有的<a>元素了吗？

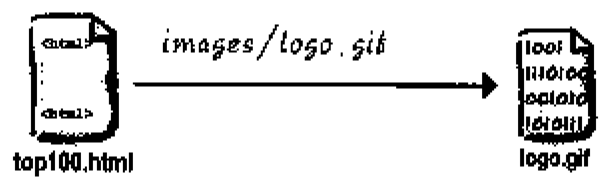


`<a href="about/directions.html" >detailed directions</a>`

把答案写在这里

相对性的重大挑战答案

Round One



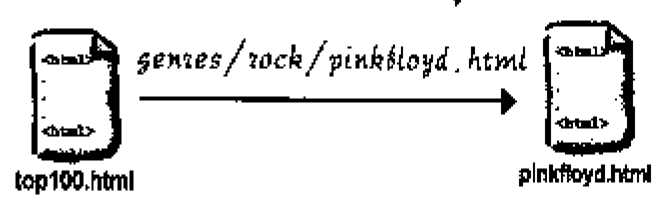
top100.html在music文件夹，我们需要下探到images文件夹来找到logo.gif。

Round Two



genres.html在genres目录下，我们先要上溯到music，再下探到image目录来找到logo.gif。

Round Three



从top100.html下探到genres，然后继续下探到rock，找到pinkfloyd.html。

Bonus Round



这个题目很阴险。从rock目录下的coldplay.html开始，我们需要上溯两个目录到达music文件夹，然后下探到images，最后找到chris.gif图像。

3 构建模块

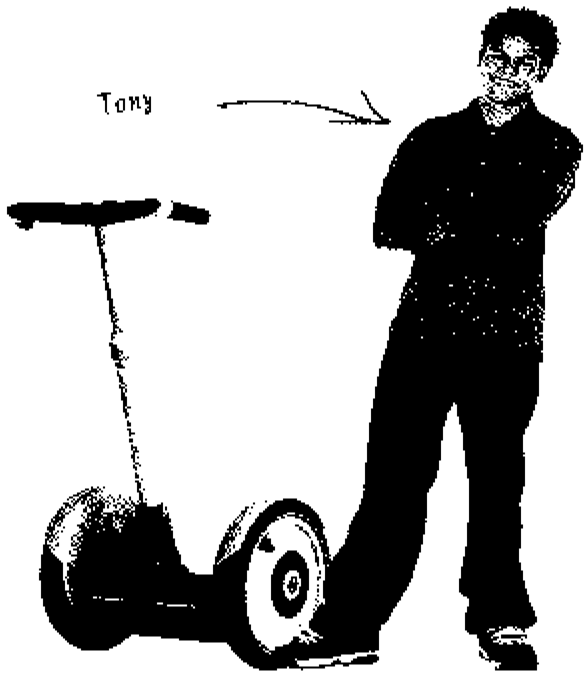
网页创建

Betty，我们最好找些安全帽戴上。这里可是真正的建筑工地，所有网页都在飞快地搭建中！



听说我能从这本书中学到如何创建网页？ 你已经学会很多了：标记、元素、链接、路径……但是如果不能学以致用，那一切都是徒劳。在这一章我们要架构建筑：你要把网站由概念变成蓝图，浇注根基，构建它，再添加些格调。你的任务是做好准备，因为我们将引入些新工具，帮你学习更深入的知识，使你工作起来得心应手。

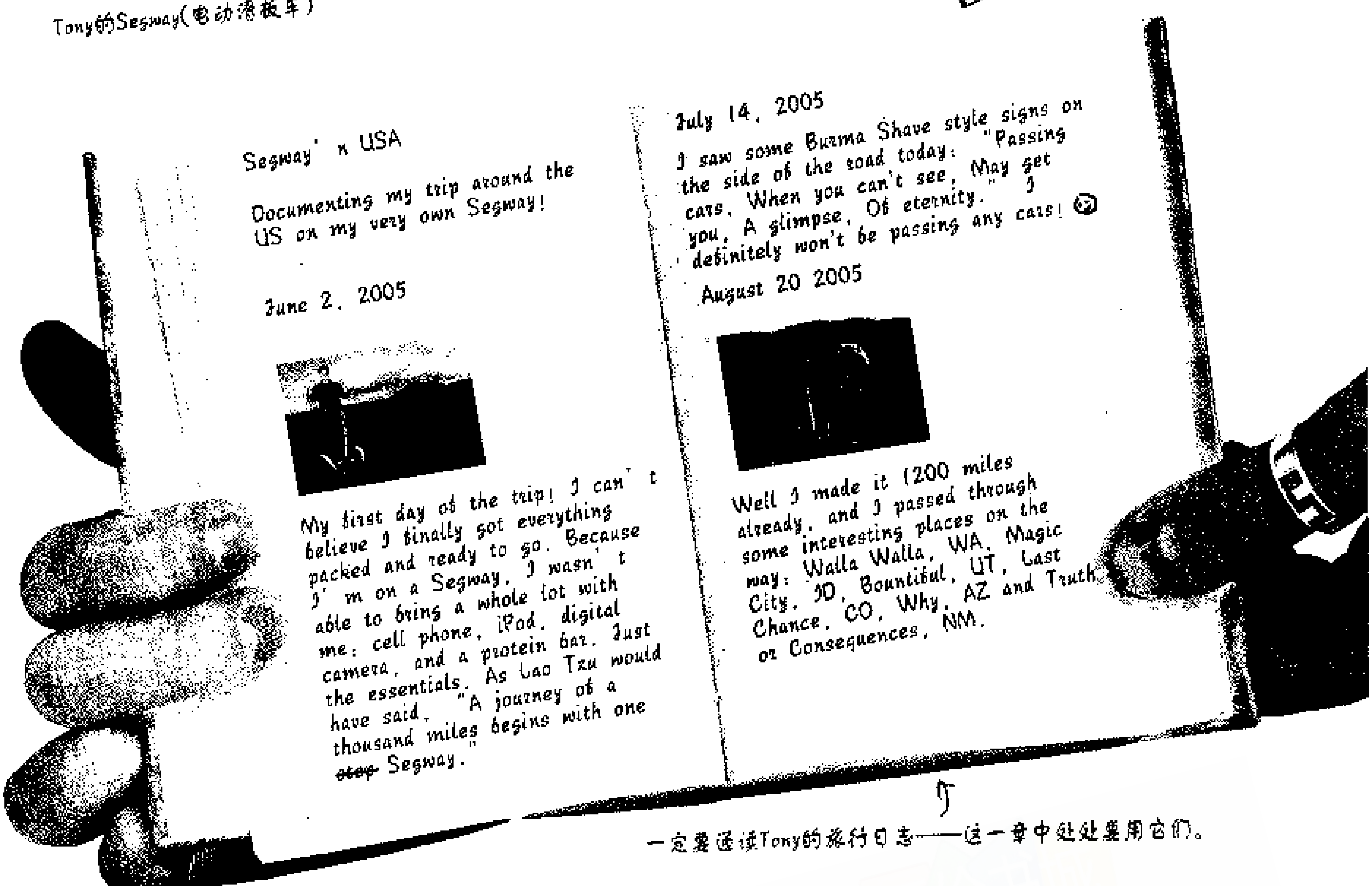
Tony



玩Segway(电动滑板车)，有比野外更好的地方吗？我骑着它穿越了整个美国，还把路上的所见所闻都记载了下来。我现在要做的是把它们放到网页上，与我的朋友家人一起分享。

Tony的Segway(电动滑板车)

Tony的旅行日志



Segway'n USA

Documenting my trip around the US on my very own Segway!

June 2, 2005

My first day of the trip! I can't believe I finally got everything packed and ready to go. Because I'm on a Segway, I wasn't able to bring a whole lot with me: cell phone, iPod, digital camera, and a protein bar. Just the essentials. As Lao Tzu would have said, "A journey of a thousand miles begins with one ~~step~~ Segway."

July 14, 2005

I saw some Burma Shave style signs on the side of the road today: "Passing cars, When you can't see, May get you, A glimpse, Of eternity." I definitely won't be passing any cars! ☺

August 20 2005

Well I made it 1200 miles already, and I passed through some interesting places on the way: Walla Walla, WA, Magic City, ID, Bountiful, UT, Last Chance, CO, Why, AZ and Truth or Consequences, NM.

一定要通读Tony的旅行日志——这一章中处处要用它们。

以每小时12公里的速度启动网站之旅

这是Segway的最高时速

Tony骑着他的Segway穿越美国，收获了许多，你就帮他建个网页吧。

你的任务是：

1. 首先，大致浏览一遍旅程，它是你网页设计的基础。
2. 然后，用HTML的基本模块（<h1><h2><h3><p>之类）把它翻译绘制成HTML能识别的略图（或者说是蓝图）。
3. 有略图后，翻译成为成型HTML。
4. 基本页完成后再做些改进，在这个过程中，你还会遇到一些新的HTML元素。

Sharpen your pencil

停！先完成这个练习再翻到下一页。

认真看看Tony的日志并思考如何把它放到网页上。

在右边画出网页的草图，假设Tony的日志条目能在一页中得以表达。不用太精致，只要绘制出大概。

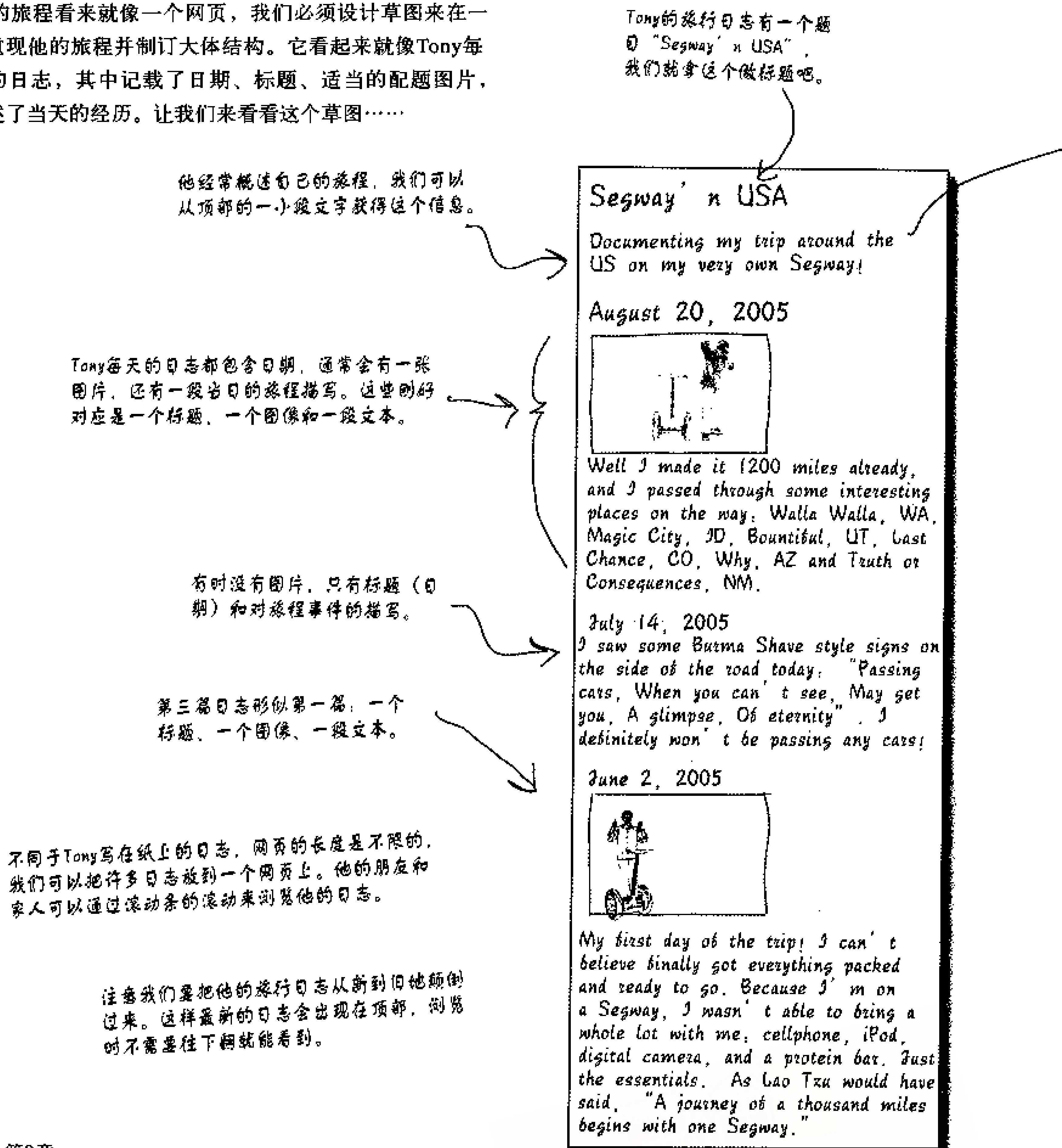
思考：

- 页面怎么用标题、主体、图像等这些结构元素构建呢？
- 该用什么方法把他的日志转变为网页呢？

你的草图画在这里

粗略的设计草图

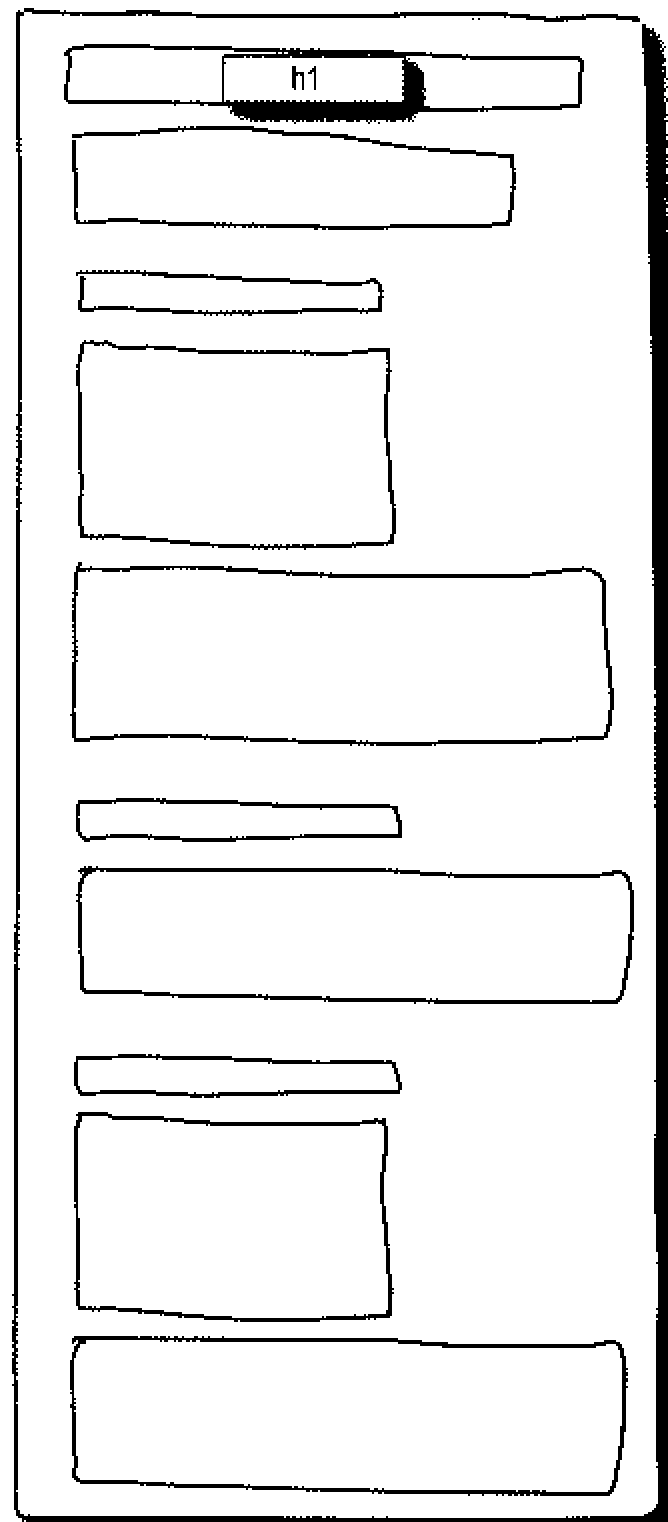
Tony的旅程看来就像一个网页，我们必须设计草图来在一页中重现他的旅程并制订大体结构。它看起来就像Tony每天写的日志，其中记载了日期、标题、适当的配题图片，还描述了当天的经历。让我们来看看这个草图……



从草图到略图

你有网页的草图后，就可以把各部分画得更接近于HTML的略图或者蓝图了。

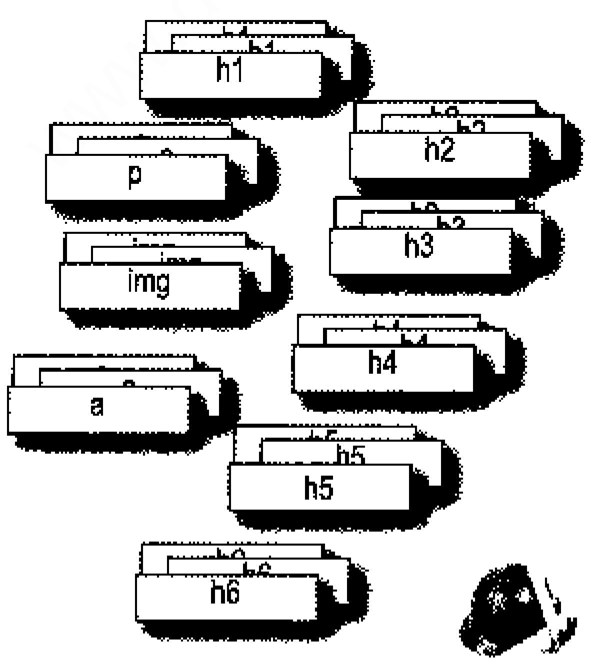
在这里我们把草图转化为相应的略图结构。



你现在所要做的就是指出哪个HTML元素对应哪块文字，然后就可以开始编写HTML了。

练习：网页构建

你已经画出了网页的主要建筑区域，现在只需要把建筑材料钉在相应的方框中就可以了。使用以下的元素来标记每个区域。你无须全部使用，不必担心建筑材料有剩余。还有，别忘了戴上安全帽。

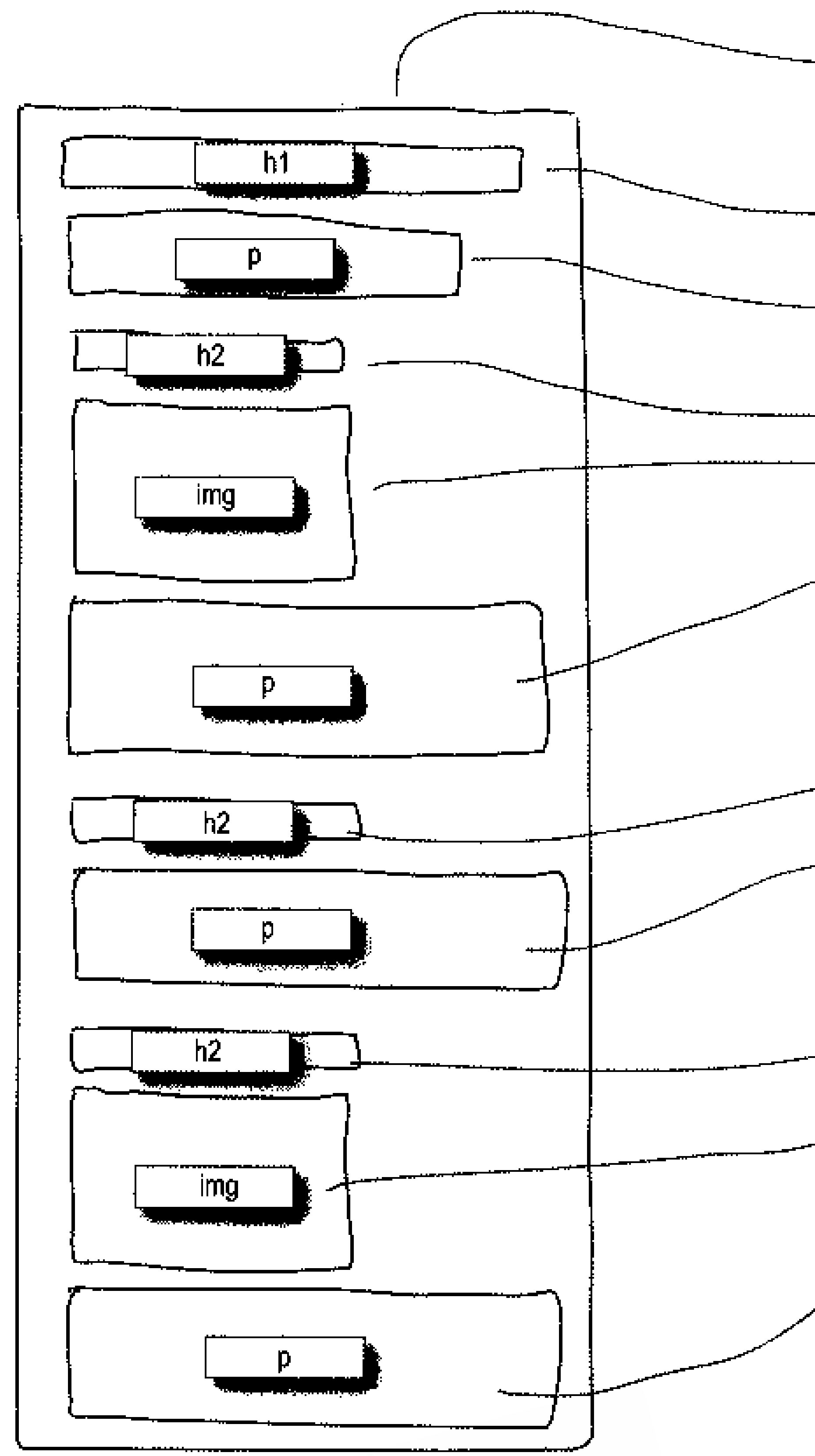


从略图到网页

胜利在望了。你已经创建了Tony日志网页的略图。现在你要做的是编写相应的HTML来显示网页，并把Tony的文字填上去。

开始之前，记住每个网页都需要以**<html>**元素开始并包括**<head>**和**<body>**两个元素。





别忘了<html>、<head>、<title>和<body>元素是不可缺少的。

以旅行日志的题目作为网页的题目。

```
<html>
  <head>
    <title>My Trip Around the USA on a Segway</title>
  </head>
  <body>
```

这是标题及对Tony旅程的描写。

```
    <h1>Segway'n USA</h1>
    <p>
       Documenting my trip around the US on my very own Segway!
    </p>
```

标题

图片

描写

这是Tony的最新日志。

```
    <h2>August 20, 2005</h2>
    <img src="images/segway2.jpg">
    <p>
      Well I made it 1200 miles already, and I passed
      through some interesting places on the way: Walla Walla,
      WA, Magic City, ID, Bountiful, UT, Last Chance, CO,
      Why, AZ and Truth or Consequences, NM.
    </p>
```

这是他的第二篇日志，没有图片。

```
    <h2>July 14, 2005</h2>
    <p>
      I saw some Burma Shave style signs on the side of the
      road today: "Passing cars, When you can't see, May get
      you, A glimpse, Of eternity." I definitely won't be passing
      any cars.
    </p>
```

底部是Tony的第一篇日志和图片"segway1.jpg"。

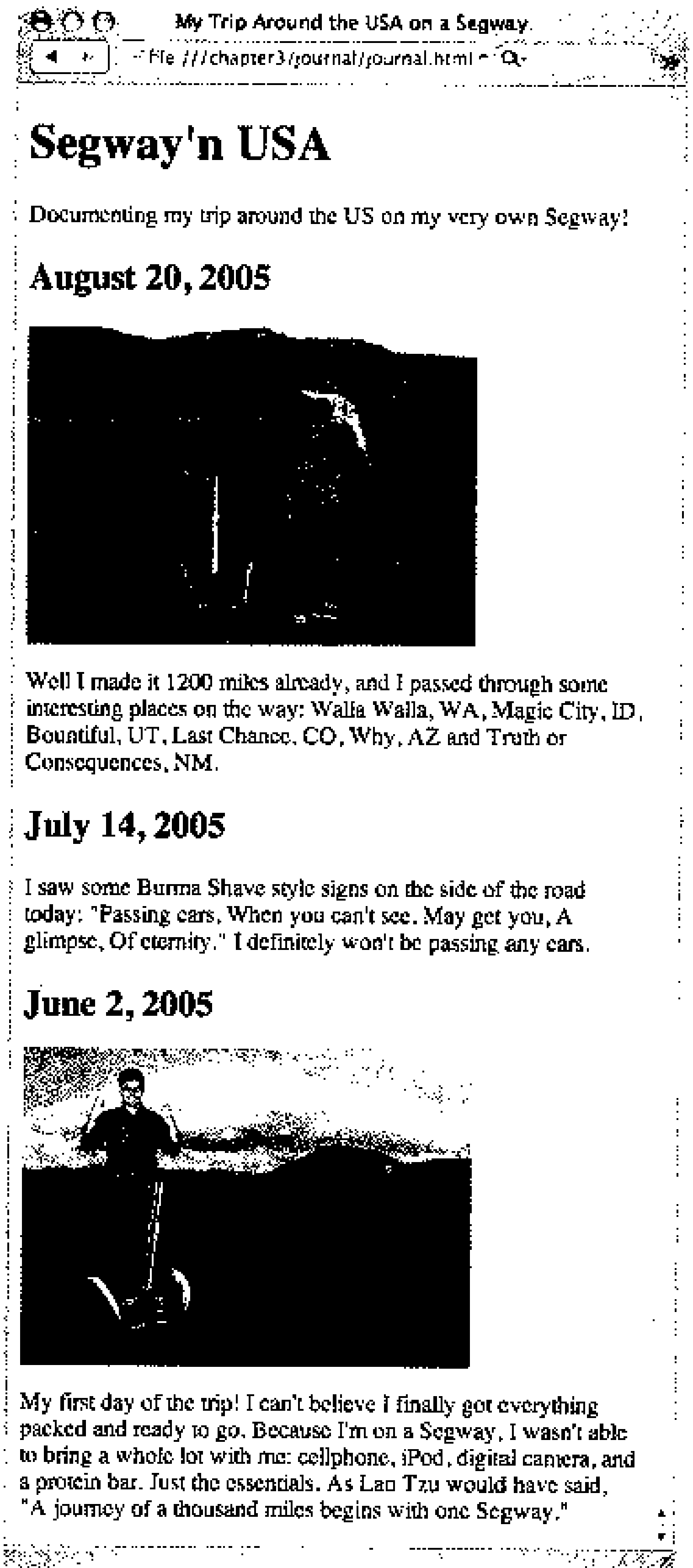
```
    <h2>June 2, 2005</h2>
    <img src="images/segway1.jpg">
    <p>
      My first day of the trip!  I can't believe I finally got
      everything packed and ready to go.  Because I'm on a Segway,
      I wasn't able to bring a whole lot with me: cellphone, iPod,
      digital camera, and a protein bar. Just the essentials.  As
      Lao Tzu would have said, "A journey of a thousand miles begins
      with one Segway."
    </p>
```

最后，别漏了结束<body>和<html>元素。

```
  </body>
</html>
```

把它输入电脑，在"chapter3/journal"文件夹中保存为"journal.html"，你会在"images"文件夹中找到"segway1.jpg"和"segway2.jpg"。完成后，测试该页。

测试Tony的网页


My Trip Around the USA on a Segway

file:///chapter3/journal/journal.html

Segway'n USA

Documenting my trip around the US on my very own Segway!

August 20, 2005

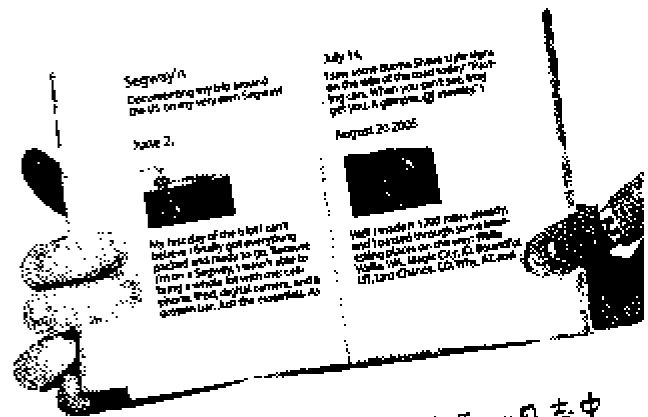
Well I made it 1200 miles already, and I passed through some interesting places on the way: Walla Walla, WA, Magic City, ID, Bountiful, UT, Last Chance, CO, Why, AZ and Truth or Consequences, NM.

July 14, 2005

I saw some Burma Shave style signs on the side of the road today: "Passing cars, When you can't see, May get you, A glimpse, Of eternity." I definitely won't be passing any cars.

June 2, 2005

My first day of the trip! I can't believe I finally got everything packed and ready to go. Because I'm on a Segway, I wasn't able to bring a whole lot with me: cellphone, iPod, digital camera, and a protein bar. Just the essentials. As Lao Tzu would have said, "A journey of a thousand miles begins with one Segway."



看看网页多漂亮，你把Tony日志中的所有内容变成了可读性强且结构严谨的网页。

非常好！看起来很不错，我已经迫不及待地想在里边添加更多日志。

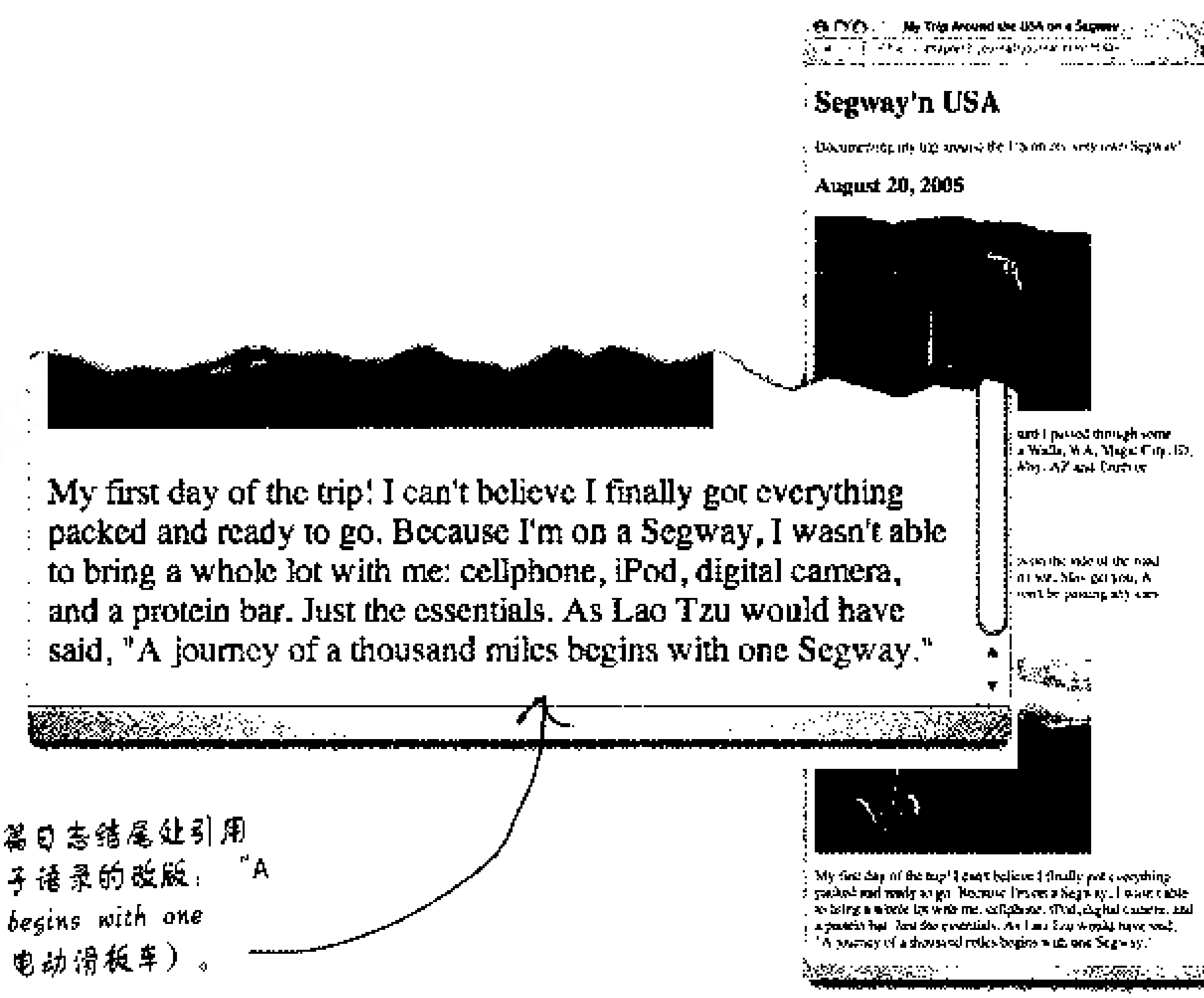


Tony正在从路上打来电话……

添加些新的元素

HTML的基本元素都有了。你用最基本的HTML元素**<p>**、**<h1>**、**<h2>**和**<img>**，经过几个步骤就把手写旅行日志变成了网上版本。

现在我们要稍微拓展一下你的思维，添加些常见的新元素。我们再来看一下Tony的日志，想想哪里可以改进。



看看这里：Tony在他的第一篇日志结尾处引用了一小段文字。这是他对老子语录的改版："A journey of a thousand miles begins with one Segway"（千里之行，始于电动滑板车）。

HTML有一个<q>元素，就是为引用文字准备的。翻到下页看看……

认识<q>元素

想在HTML中来段简短的引用吗？<q>元素正是你需要的。下面是一个HTML的小演示，向你展示它是如何工作的。

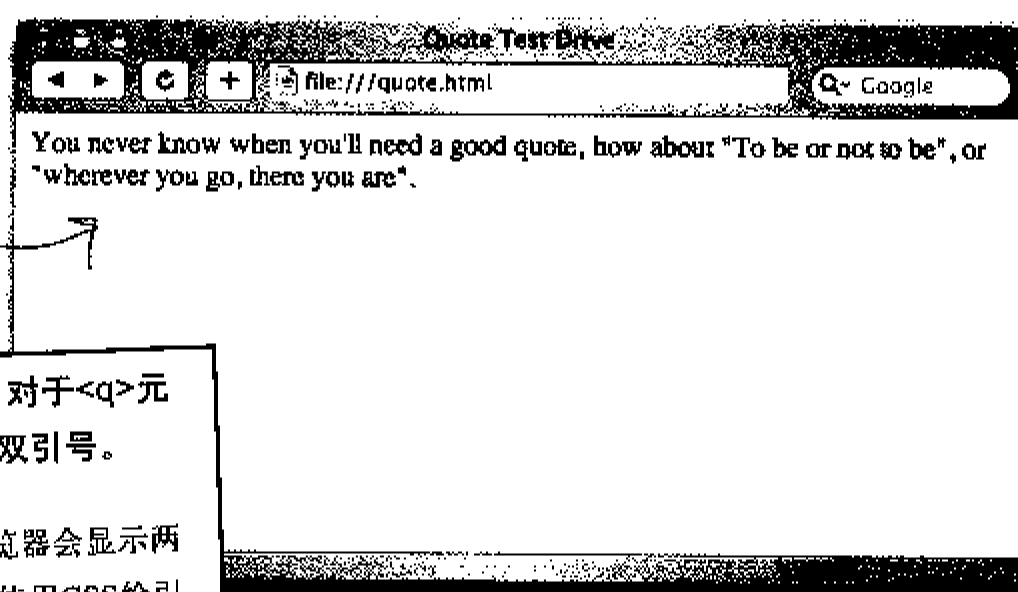
```
<html>
  <head>
    <title>Quote Test Drive</title>
  </head>
  <body>
    <p>
      You never know when you 'll need a good quote, how
      about <q>To be or not to be</q>, or <q>Wherever you go, there you are</q>.
    </p>
  </body>
</html>
```

在这段HTML中有两个引用……

我们用<q>开始标记和</q>结束标记来包围每段引用。注意使用双引号显示会不正常。

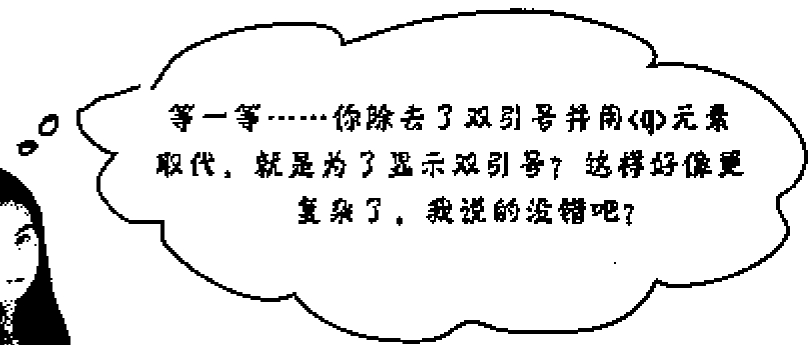
……然后进行测试……

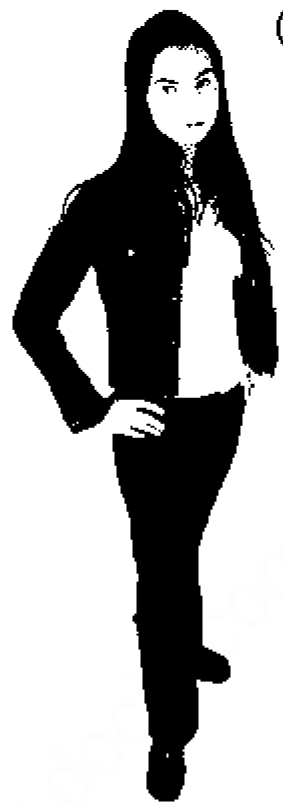
这里显示引用的效果。注意浏览器会因为你输入双引号而出现问题。



注意！ **某些浏览器，包括IE6.0，对于<q>元素包围的文字周围不显示双引号。**

真不幸，如果你加入了双引号，某些浏览器会显示两组引号。解决这个难题的唯一方法就是使用CSS给引用添加视觉样式，例如斜体字。我们会在第 9 章教你如何给元素添加斜体字。





不是，我们想把事情变得更结构化并使其更有意义。

人们在文本中使用双引号的原因有很多，但是我们使用**<q>**就意味着某些特殊的东西——即文本的真正引用（在Tony的例子中，是个“改写版本”）。

看，使用双引号不代表真正引用。

换句话说，我们用标记引用的方法来添加额外的信息。在添加**<q>**元素之前，浏览器只知道一段文字和一些双引号在里边。现在，因为我们使用了**<q>**元素，浏览器就能知道哪些文本才是真正的引用。

这有什么意义呢？看，是这样的，当浏览器知道这是个引用，就会尽量用最好的方式显示它。某些浏览器会在文字周围显示双引号，某些不会，比如在浏览器没有使用英语时，就要用其他方法。不要忘记移动设备，像手提电话，或者为视觉有障碍人士准备的HTML视频浏览器。这在其他场合也很有用，例如搜索引擎寻找带引用的网页。在你的网页中结构和方法是很重要的。

最重要的一个原因（在本书后面当我们回到外观和CSS的时候，你会看到）是你能按你的意愿样式化引用。假如你想引用一段文字，并用灰色的斜体字显示，使用**<q>**元素组织网页中引用的内容，就能做到。

练习

这是Tony的日志。用<q>元素重写他引用的老子的话。在纸上做完以后，更改journal.html并测试。你可以在本章后面找到答案。

```
<html>
  <head>
    <title>Segway 'n USA</title>
  </head>
  <body>

    <h1>Segway'n USA</h1>
    <p>
        Documenting my trip around the US on my very own Segway!
    </p>

    <h2>August 20, 2005</h2>
    <img src="images/segway2.jpg">
    <p>
      Well I made it 1200 miles already, and I passed
      through some interesting places on the way: Walla Walla,
      WA, Magic City, ID, Bountiful, UT, Last Chance, CO,
      Why, AZ and Truth or Consequences, NM.
    </p>

    <h2>July 14, 2005</h2>
    <p>
      I saw some Burma Shave style signs on the side of the
      road today: "Passing cars, When you can't see, May get
      you, A glimpse, Of eternity." I definitely won't be passing
      any cars.
    </p>

    <h2>June 2, 2005</h2>
    <img src="images/segway1.jpg">
    <p>
      My first day of the trip!  I can't believe I finally got
      everything packed and ready to go.  Because I'm on a Segway,
      I wasn't able to bring a whole lot with me: cellphone, iPod,
      digital camera, and a protein bar. Just the essentials.  As
      Lao Tzu would have said, "A journey of a thousand miles begins
      with one Segway."
    </p>
  </body>
</html>
```

双胞胎奇案

许多年前，在Web镇诞生的双胞胎因为一个奇异的事故——因特网路由器故障，在出生后就分开了。在不知道对方存在的情况下长大，发生了一系列匪夷所思的情况后，他们相遇并发现对方的身份，他们决定不公开这个秘密。

五分钟之谜



此后，他们发现彼此有着惊人的相似之处。他们都娶了个名字叫Citation的妻子；都喜欢引用。双胞胎之一，<q>元素，喜欢精练简短的引用；而另一个，<blockquote>则喜欢块引用，他经常记下书上或者诗歌中的整段文字。

作为双胞胎，他们在歪主意方面也有很多相似之处，他们制定了一个邪恶的计划，决定从今以后互相代替对方。他们先拿自己的妻子做实验（细节我们就不详述了），他们成功了！妻子们都没发觉（或者至少是假装没发觉）。

接着他们交换工作，另外一个巧合是，他们都做同样的工作：在HTML文件中标记引用。某一天，两兄弟满怀信心地到对方的工作场所执行他们荒诞的计划（毕竟，连他们的妻子都无法辨认，更何况老板呢？），这时，形势突变。10分钟以后，两兄弟都被发现是冒名顶替的，并马上被报告给了安全权威机构。

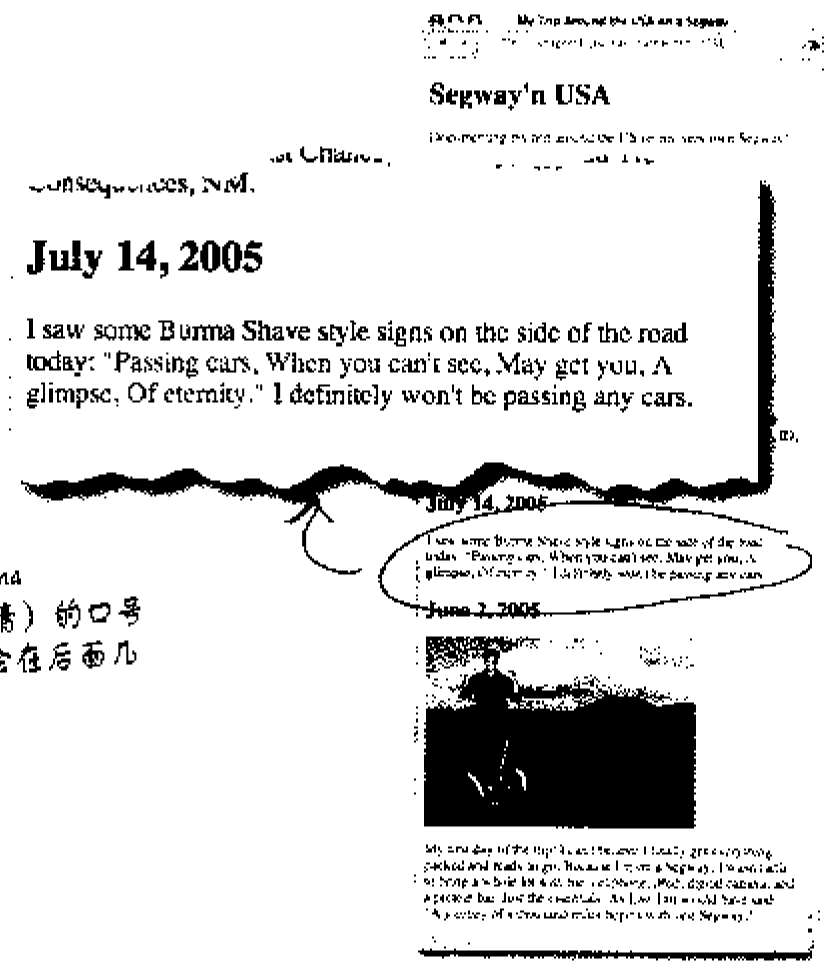
为什么双胞胎会在这起案件中事发呢？

继续读下去，你会发现更多线索……

www.docin.com

很长很长的引用

现在你知道怎么应付短引用了，我们来处理些块引用吧。Tony给了个关于柏玛刮胡膏广告的长引用。



在旅行日志中，Tony把对柏玛刮胡膏广告的引用放到段落中，如果我们把这个引用单独做成“块”会不会更好一些呢？就像这样：

I saw some Burma Shave style signs on the side of the road today:

> *Passing cars,*
> *When you can't see,*
> *May get you,*
> *A glimpse,*
> *Of eternity.*

I definitely won't be passing any cars.

如果你不知道“Burma Shave”（柏玛刮胡膏）的口号是什么的话，我们会在后面几页告诉你。

<blockquote>该发挥作用了。<q>元素是简短地“引用”现有文字中的一部分，而<blockquote>则是引用一大段文字并独立显示。



添加<blockquote>

让我们在Tony的网上日志中添加<blockquote>吧。

1. 打开"journal.html"文件并找到7月14号的日志。改写编码如下：

```
<h2>July 14, 2005</h2>
<p>
   I saw some Burma Shave style signs on the
   side of the road today:
</p>
<blockquote>
   Passing cars,
   When you can't see,
   May get you,
   A glimpse,
   Of eternity.
</blockquote>
<p>
   I definitely won't be passing any cars.
</p>
```

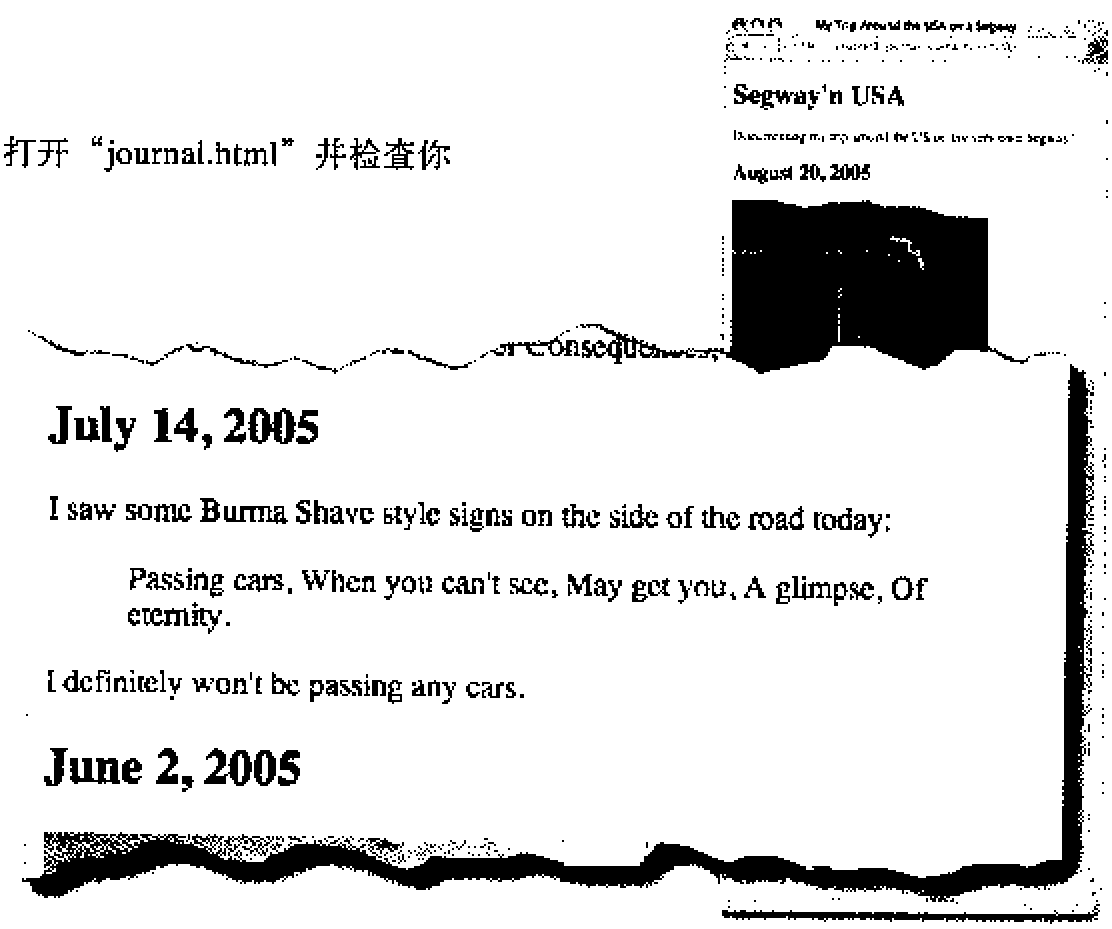
段落结束之后才能插入<blockquote>元素。

把构瑪剃胡膏的广告词放到<blockquote>元素中。

我们把每一句单独放一行，使之看起来更像构瑪剃胡膏的口号。

最后，添加<p>标记，在<blockquote>后开始这个段落。

2. 做另外一个测试。在浏览器中打开"journal.html"并检查你的成果。



<blockquote>创建了一段单独的文字（功能如同<p>标记），另外，文本有缩进，显得更像一段引用的文字。这正是我们想要的效果。

但是引用看起来和我们的目标有差距，所有的行都挤到一块了。我们希望它们分行显示。嗯，继续努力吧。

there are no Dumb Questions

问： **看看我是不是可以这么做：网页中，如果我想在文字段中少量引用的话就用<q>，如果需要另开一段来进行引用则用<blockquote>？**

答： 你说得对。通常希望引用一段或者更多文字的时候使用<blockquote>，而希望在文字中夹杂一点引用的话就用<q>。

问： **假如我要引用几段文字呢？应该怎么办？**

答： 这个容易。用<blockquote>包围每段文字，一段一个。你可以回家试试看。

问： **怎么知道我的短引用或块引用在别的浏览器中所显示的效果呢？听起来它们的处理方式好像有所不同。**

答： 对。欢迎来到万维网。不尝试就无法知道你的引用在别的浏览器中是如何显示的。某些浏览器使用双引号，某些使用斜体字，还有些什么都不使用。能确定他们如何显示的唯一方法是样式化引用，我们以后再解决这个问题。

问： **<blockquote>可以使文章另开一段并缩进显示，但为什么它不像<q>那样引用在段落内呢？**

答： 因为<blockquote>其实像一个新的段落。想象你正在往文字处理器中输入一篇文章。完成一个段落后，你敲两次空格键来开始一个新的段落。使用块引用会有相同的缩进效果。暂时别去想它吧。这是个要点，我们以后会回头来研究它。

同样，缩进是某些浏览器显示<blockquote>的效果。不是所有的浏览器对<blockquote>都缩进，也有可能在新版本中有所变化。所以，不要指望<blockquote>在所有的浏览器中显示一致。

问： **引用元素可以一起使用吗？比如，可以把<q>元素放到<blockquote>中吗？**

答： 可以。就像可以把<q>放到<p>元素中一样，你也可以把<q>放到<blockquote>中。如果你想引用别人的引用过的话，可以这么做。不过把<blockquote>放到<q>中就没有任何意义。

问： **你说可以用CSS来样式化元素，那如果我想用灰色的斜体显示<q>元素中的文本，就可以用CSS来做。但是能不能用<em>元素来使我的引用用斜体字显示呢？**

答： 嗯，可以，但这不是最好的方法，因为你使用<em>元素只是为了达到显示效果而不是真的要强调这段文字。如果你引用的人真的说了句很值得重视的话，或者你想添加强调符号来告诉别人这里是重点，那就在你的引用里边使用<em>元素吧。但是不要以这种方式获得斜体字。使用CSS比较容易获得理想的外观而且效果更好。

答案：双胞胎奇案。

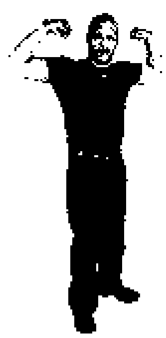


为什么双胞胎引用那么快就被发现是冒名顶替的？

不用再猜了，**<q>**和**<blockquote>**在文本中添加标记时被发现了。**<q>**本该经常引用不醒目的文本，现在却突出出来，独占一块；而**<blockquote>**段落则在平常的文本中突然消失。在之后对这个恶作剧的受害者的采访中，一位编辑抱怨说："这两个疯子害得我整个网页都是分行引用。"在被申诉并回到各自的岗位后，**<q>**和**<blockquote>**向他们的妻子坦白了，她们随即驾车离开了小镇。这就是案件的始末（尽管结局不太理想）。

<q>和<blockquote>奇案背后的真相

好了，现在是时候结束字谜了：**<blockquote>**和**<q>**是两种不同类型的元素。**<blockquote>**元素是块（block）元素而**<q>**元素是内联（inline）元素。它们有什么区别呢？块元素前后都有换行符，而内联元素总是在网页中随着文字流出现在“行内”。



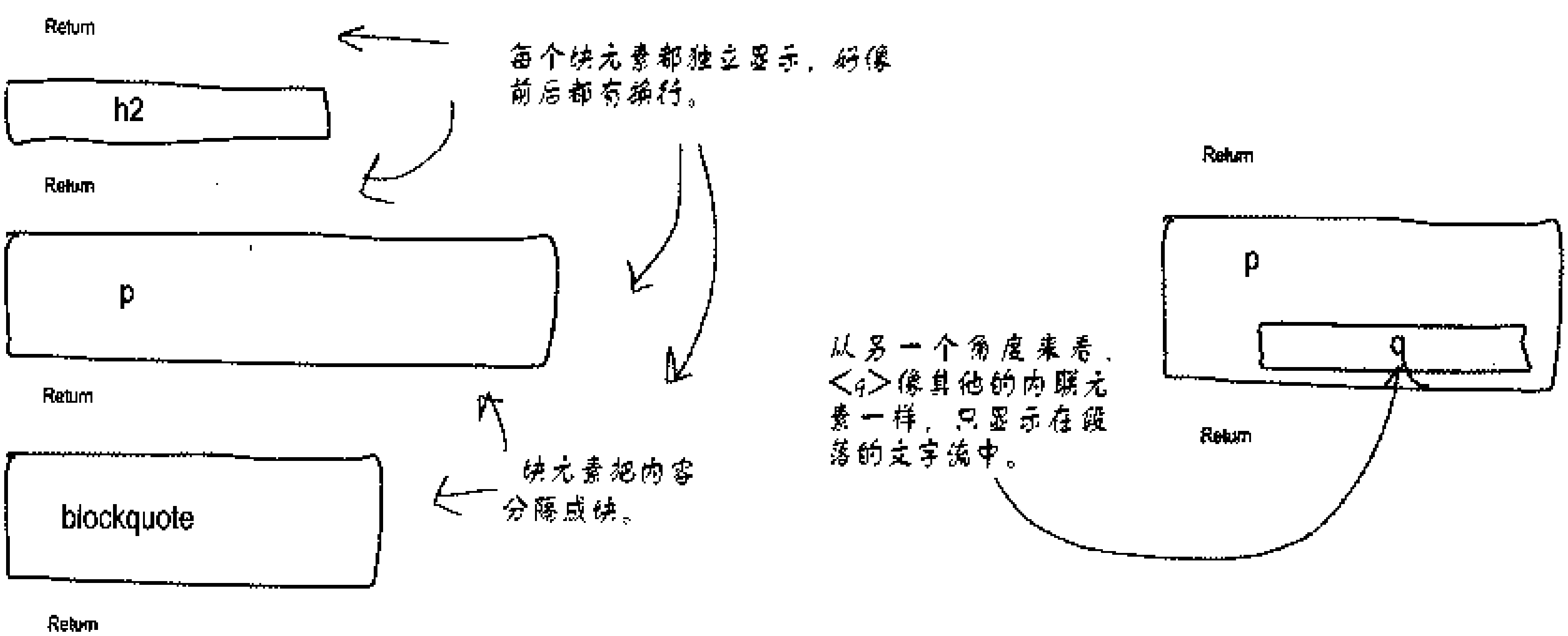
块：特立独行



内联：随波逐流

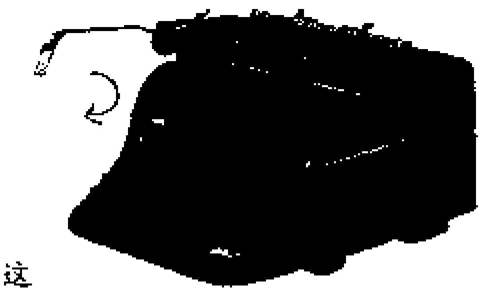
<h1><h2>,……,<h6>和<blockquote>都是块元素。

<q><a>和<em>都是内联元素。



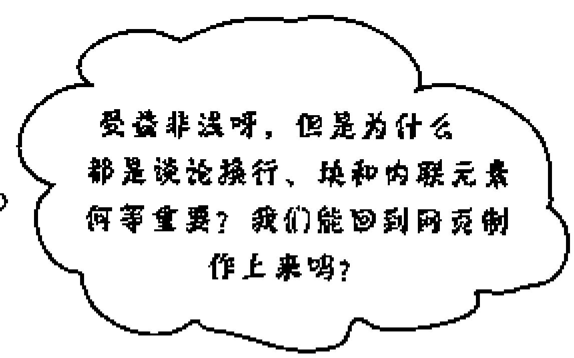
记住：块元素特立独行；内联元素随波逐流。

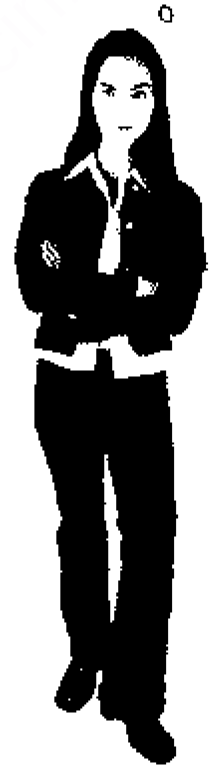
there are no Dumb Questions



问： 我想我知道什么是换行了，就像在打字机或者电脑键盘上按return键。对不对？

答： 基本上是。照字面意思来说换行就是换到新的一行，就像这样、 按return键就是这种效果，在某些电脑上，是回车键。你已经知道浏览器显示页面时，HTML文件中的换行是不显示的，对不对？但是现在你已经看到无论什么时候用块元素，浏览器都会用换行来分隔每个“块”。



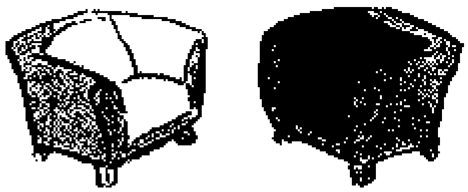


不要低估了了解HTML如何工作的重要性。你很快就会看到，在网页中结合使用元素的方式和该元素是块还是内联之间有很大关系。我们稍后再研究这个。

现在，你可以研究块和行：块元素是网页的主要结构模块，而内联元素则用来标记内容的小片段。当你设计网页时，开头会面对一大块东西（块元素），然后添加内联元素来修饰网页。

当我们用CSS来控制HTML的外观的时候，刚才的知识就会派上用场了。你知道块元素和内联元素之间的区别，就可以轻松布置好外观了，然后悠闲地喝着马提尼酒，而其他人可能还在为布局忙活呢。

围炉夜话



今夜话题：内联元素和块元素公布他们的不同

内联元素

块元素，欢迎你。你出现在这里真让我有点吃惊。

块元素

为什么？

内联元素

因为你是个孤独的人。你经常让换行围在你周围以此与别人保持距离。他们看起来就像你的贴身保镖或者其他什么。

块元素

我是个很忙的人。块元素是网站的主要构建模块。如果没有我，网页会崩溃的。

内联元素

不要把自己想得那么伟大。对，你很重要，但是没有内联元素在内你能怎么样？没有文本和内联内容，比如链接，主体和标题都是没意义的。

块元素

我同意<a>是个重要的元素，我们都试图把他拉到我们这边来。但是，网页最重要的部分是在建立时就设计好的，这要靠块元素。你不能拿一堆链接来做一个网页，对不？

内联元素

我会告诉你<a>不会离开我的。他生是内联元素的人，死是内联元素的鬼。而且如果你的网页没有<a>、<em>、<q>元素和其他内联元素，就算根基打得再好，也做不出有趣的网页。

内联元素

块元素

我们可能没有<a>那样的功效，但是我们已经让<img>在我们这边工作了好几年了。他会是个优秀的块元素。

好，许多人认为<img>是个块元素，其实不是，他是作为一个内联元素发挥着巨大的作用。人们喜欢把图片融入到他们的文本和链接中。

这个以后再说。我会告诉你另外一个事实，<blockquote>相对于<q>来说好多了。我们已经有一个完美的块引用，为什么还需要<q>呢？

因为人们喜欢在文章内使用小巧的引用。我没有对<blockquote>说什么，为什么你对<q>那么挑剔？要知道，你错在招收了许多内联元素却认为他们不太重要。

换行保镖们呢？站那么远干什么！我要回去创建网页了。

哦，真方便。让我看看没有任何内联元素你怎么创建网页吧。我敢肯定，不会做出什么有用的网页。肯定不能！



假如你需要换行的时候发现有个元素唯一的功用恰恰就是换行，你会怎么想？

那不是很好吗？这样你就能提醒浏览器注意并插入些换行。

揭晓答案吧，这就是**
**元素，它正是为这个目的而诞生的。以下教你如何运用它：

这是Tony的网页中7月14号日志的片断。

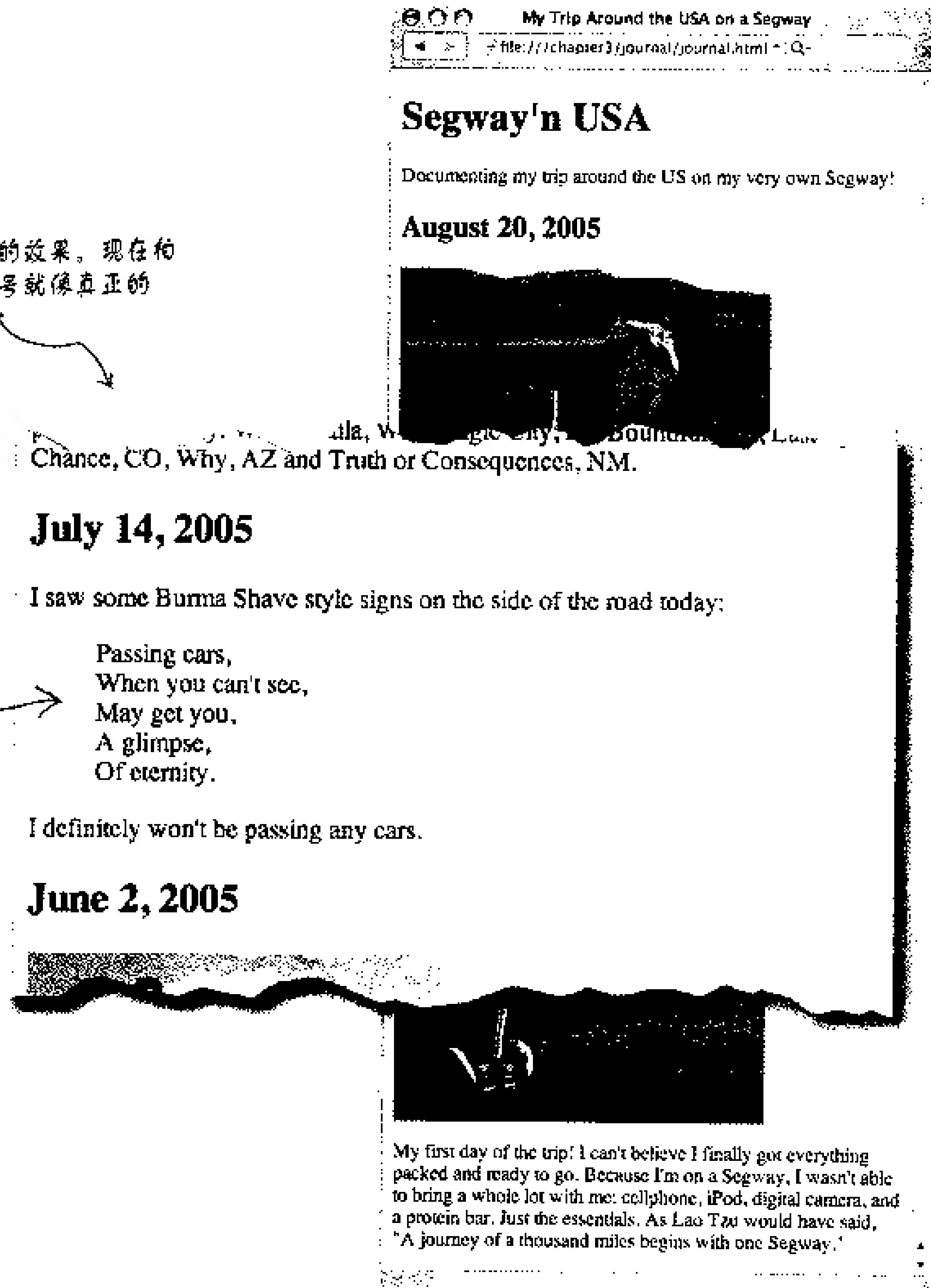
```
<h2>July 14, 2005</h2>
<p>
   I saw some Burma Shave style signs on the
   side of the road today:
</p>
<blockquote>
   Passing cars, <br>
   When you can't see, <br>
   May get you, <br>
   A glimpse, <br>
   Of eternity. <br>
</blockquote>
<p>
   I definitely won't be passing any cars.
</p>
```

在你需要换行的地方加
元素。

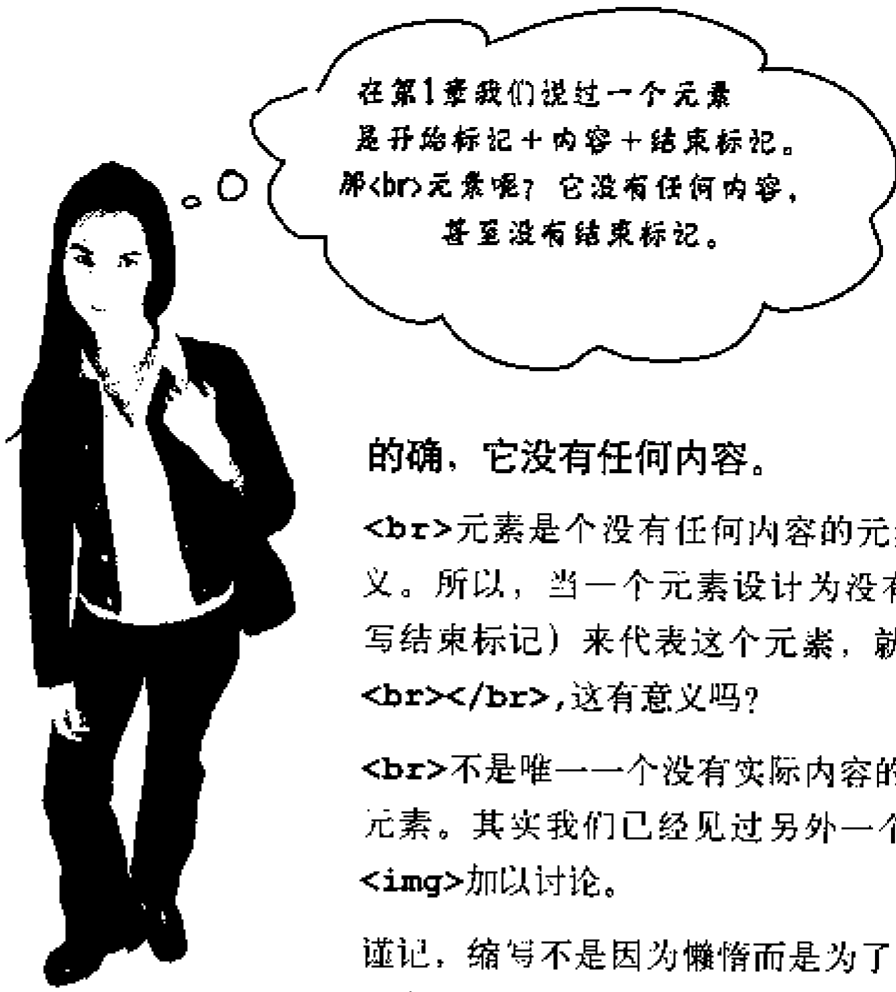
练习

试着把
元素加到Tony的旅行日志中。完成后，保存并测试。

这里是修改后的效果，现在伯瑞利刮青的口号就像真正的口号了。



每一行都有换行。

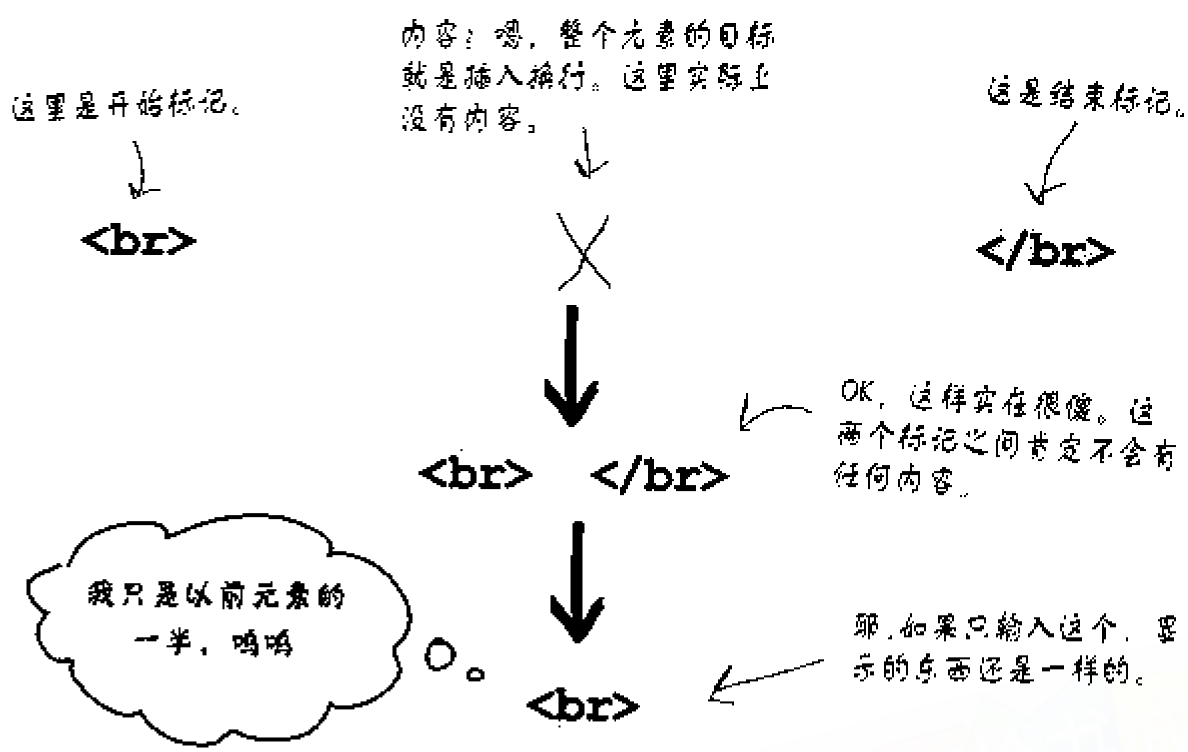


的确，它没有任何内容。

**
元素是个没有任何内容的元素。为什么？因为它只代表换行，没有其他意义。所以，当一个元素设计为没有任何实际内容的时候，我们就使用缩写（不写结束标记）来代表这个元素，就像
一样。毕竟，如果不缩写，换行时用
</br>**，这有意义吗？

**
不是唯一一个没有实际内容的元素。还有其他的，他们有个共同的名字：空元素。其实我们已经见过另外一个空元素了，即<img>**元素。几章后我们会对**<img>**加以讨论。

谨记，缩写不是因为懒惰而是为了有效。这样代表空元素效果更佳（体现在输入、页中的字数等上）。实际上，读完下面的HTML后，你会发现这样还可以减轻眼睛的负担。



there are no Dumb Questions

问： 那么，
的唯一目的就是插入一个换行？

答： 对。只有在开始一个新的块元素时（像<p>、<h1>之类），浏览器才会插入换行。如果想直接在文本中插入换行，就得使用
元素。

问： 为什么
被称为“空”元素？

答： 因为它没有内容。元素＝开始标记＋内容＋结束标记。因为它没有内容，所以是空的。

问： 我始终没明白。你能解释一下为什么
元素是“空的”吗？

答： 想一下<h1>(或<p>、<a>)这样的元素。它的基本任务是标记内容，例如：

<h1>Don't wait ,order now</h1>

元素的作用就是在HTML中插入换行。它没有可以标记的内容，所以是空的。因此我们没必要用上全部尖括号和标记，而是缩写成更方便的形式就可以了。

如果一个元素不需要标记文本，那它就可能是个空元素。

问： 还有其他空元素吗？我觉得<img>元素也是个空元素，对不对？

答： 对，还有其他空元素。你已经接触过<img>元素了，稍后我们会对它加以介绍。

问： 任意元素都可以为空吗？例如我想有个链接，但是不想有任何内容，那我可以写成<a href= “mypage.html” >吗？

答： 不行。世界上有两种类型的元素：一般元素，例如<p>、<h1>和<a>；还有空元素，如
和<img>。不能混用两种元素。如果你输入<a href= “mypage.html” >，那它不是一个空元素——而是个没有内容和结束标记的开始标记。

**没有HTML内容的元素称为空元素。当你需要一个空元素例如
或者<img>时，只需要写一个开始标记。这种方便的缩写减少了HTML标记的数量。**

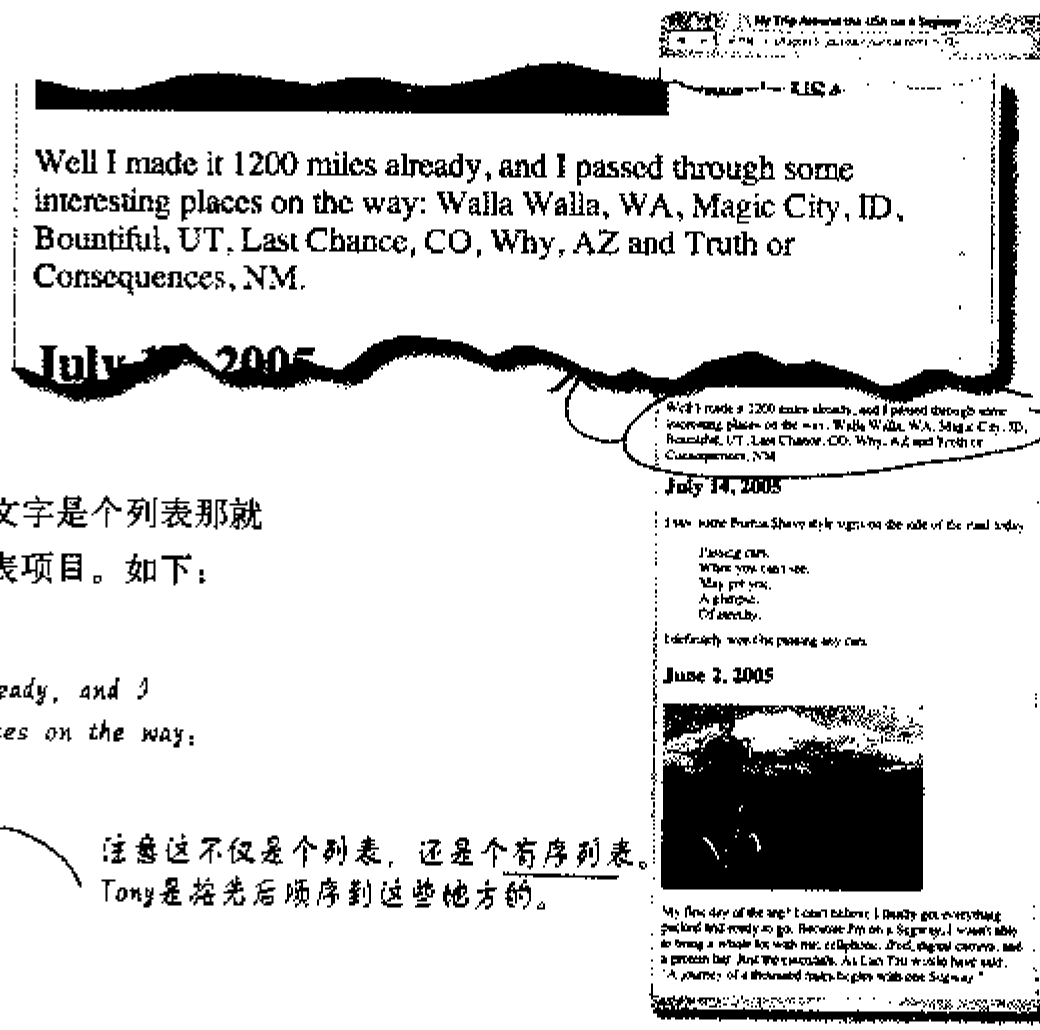
回到Tony的网站……

你在这章已经收获了许多：你已经设计并创建了Tony的网站，认识了一些新的元素，还学到了关于空元素的知识，那是许多创建网页的人都不知道的（比如块和内联元素，它们将在后面的章节中真正派上用场）。

但是你的工作还没完成。恰到好处地添加些标记，我们可以把Tony的网站从良好升华到优秀。

添加什么？列表怎么样？尝试一下：

这是一个列表。Tony用这个列表记载他在八月去过的地方。



如果我们能通过标记让浏览器知道这段文字是个列表那就太棒了。这样浏览器会更有效地显示列表项目。如下：

Well I've made it 1200 miles already, and I passed through some interesting places on the way:

1. Walla Walla, WA
2. Magic City, ID
3. Bountiful, UT
4. Last Chance, CO
5. Why, AZ
6. Truth or Consequences, NM

注意这不仅是个列表，还是个有序列表。Tony是按先后顺序到这些地方的。

当然，你能用<p>元素来做一个列表……

用<p>元素做列表也不是很难。就像这样：

```
<p>
1. Red Segway
</p>
<p>
2. Blue Segway
</p>
```

前两个为电动滑板车选颜色。

但诸多因素使我们放弃这个选择。

现在你应该得出结论了：总是要选择方法与内容结构最接近的HTML元素。如果这是个列表，就使用列表元素吧。这样会让你（稍后你会看到）和浏览器以最有效便利的途径显示内容。



BRAIN POWER

为什么不使用<p>来做列表？（多选）

- ❑ A　HTML有个专门做列表的元素。如果使用它，浏览器就知道这段文字是列表，并尽可能用最佳方式显示它。
- ❑ B　<p>元素实际上代表文本段落而不是列表。
- ❑ C　它可能看起来不像一个列表，而像一堆标了号的段落。
- ❑ D　如果你想改变列表的顺序，或者插入新项目，就必须重新给它们标号，那会很麻烦。

答案：A，B，C，和D。

用简单的两步创建HTML列表

创建一个HTML列表需要两个元素。这两个元素结合使用就构成了列表。第一个元素用来标记列表项目。第二个元素决定你创建的列表的类型：有序还是无序的。

让我们分步创建Tony到过的城市的HTML列表。

第一步：

把每个列表项目放进一个<li>元素中。

把每个列表项目放到各自的
元素中来创建列表。这意味着用开始标记
和结束标记
把所有内容封入。和其他HTML元素一样，标记之间的内容长短不限，也可以分成几行。

这里显示了Tony日志的HTML的一部分。

在你的"journal.html"找到这一段HTML并和我们一起做。

```
<h2>August 20, 2005</h2>
    <img src="images/segway2.jpg" />
<p>
Well I've made it 1200 miles already, and I passed
through some interesting places on the way:
</p>

<li>Walla Walla, WA</li>
<li>Magic City, ID</li>
<li>Bountiful, UT</li>
<li>Last Chance, CO</li>
<li>Why, AZ</li>
<li>Truth or Consequences, NM</li>

<h2>July 14, 2005</h2>

<p>
I saw some Burma Shave style signs on the side of
the road today:
</p>
```

首先把列表项移出段落。列表会独立地显示。

用
把每个列表项封入。

每个<li>元素将在列表中开始一个列表项。

第二步：

用<ol>或者<ul>封装列表元素。

如果使用<ol>元素来封装列表元素，那么它们将显示为有序列表。如果使用<ul>，列表将显示为无序列表。这里是用<ol>元素封装列表元素的例子。

这里还是显示Tony日志的HTML的一部分。

```
<h2>August 20, 2005</h2>
    <img src="images/segway2.jpg" />
<p>
Well I've made it 1200 miles already, and I passed
through some interesting places on the way:
</p>

<ol>
  <li>Walla Walla, WA</li>
  <li>Magic City, ID</li>
  <li>Bountiful, UT</li>
  <li>Last Chance, CO</li>
  <li>Why, AZ</li>
  <li>Truth or Consequences, NM</li>
</ol>


<h2>July 14, 2005</h2>

<p>
I saw some Burma Shave style signs on the side of
the road today:
</p>
```

我们想要一个有序列表，因为Tony是按特定的顺序参观这些城市的。所以需要用到开始标记<ol>。

所有的列表项均放置在<ol>元素中间，成为它的内容。

在这里我们结束<ol>元素。

BRAIN POWER

是块元素还是内联元素？<li>呢？

强化记忆

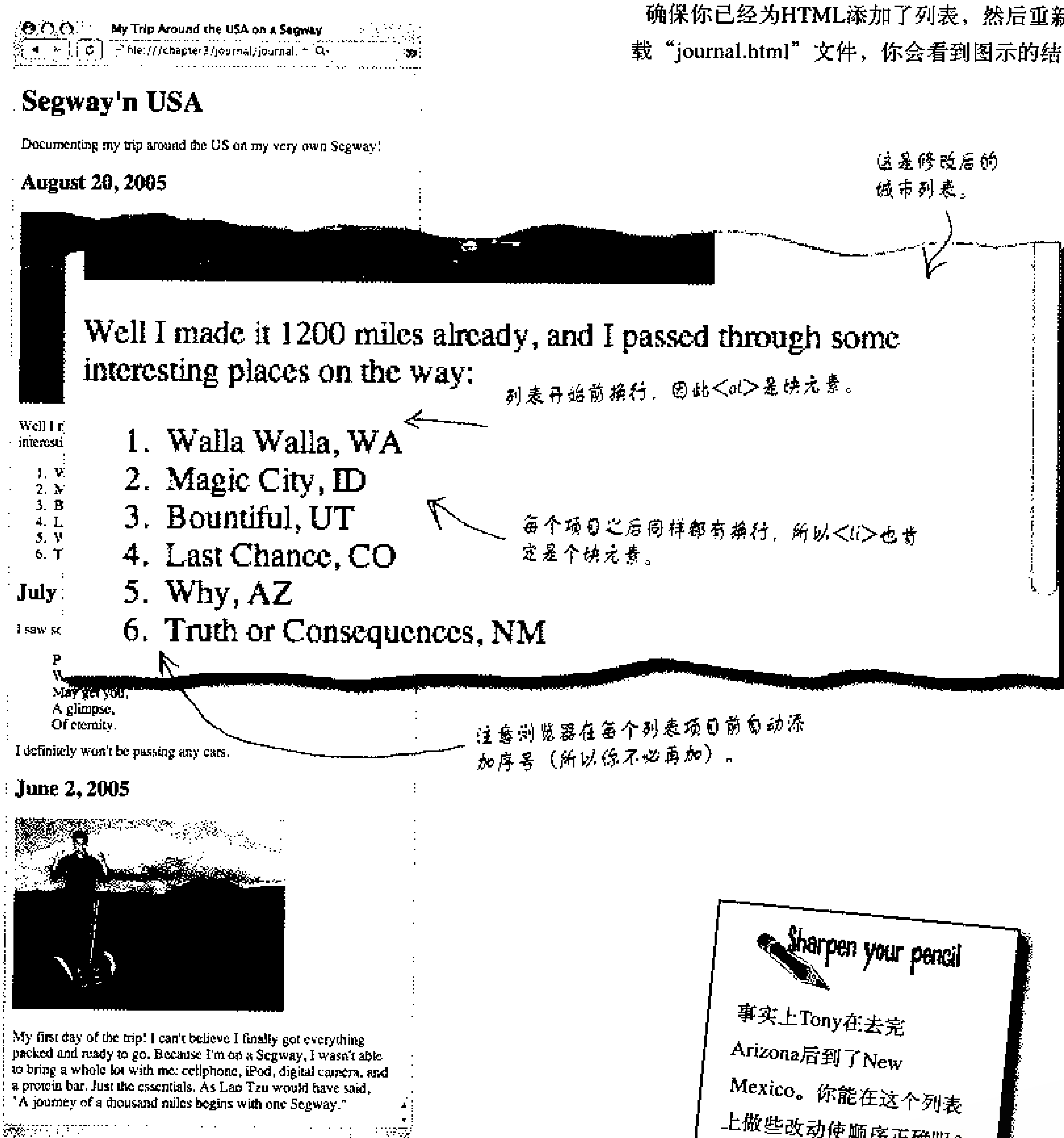
unordered list = ul

ordered list = ol

list item = li

测试城市列表

确保你已经为HTML添加了列表，然后重新加载“journal.html”文件，你会看到图示的结果：



Sharpen your pencil

事实上Tony在去完Arizona后到了New Mexico。你能在这个列表上做些改动使顺序正确吗？

练习

这里是Tony旅行日志中的另一个列表：手提电话、iPod、数码相机，还有蛋白棒。这是一个无序列表，你可以在他6月2日的日志中找到。

以下是关于这篇日志的HTML。添加HTML使上述项目变成HTML无序列表（记住，使用<ul>来创建无序列表）。我们已经重新格式化了一些内容。

做完后，对照本章后的答案。然后在“journal.html”中修改并测试。

```
<h2>June 2, 2005</h2>
<img src="segway1.jpg">
<p>
  My first day of the trip!  I can't believe I finally got
  everything packed and ready to go.  Because I'm on a Segway,
  I wasn't able to bring a whole lot with me:


      cell phone
      iPod
      digital camera
      and a protein bar


  Just the essentials.  As
  Lao Tzu would have said, <q>A journey of a
  thousand miles begins with one Segway.</q>
</p>
```

there are no Dumb Questions

问： **<ol>和<li>必须一起使用吗？**

答： 对，你必须一起使用<ol>和<li>（或者<ul>和<li>）。它们唇齿相依，缺一不可。记住，列表是一组项目：<li>元素用来确定每个项目，<ol>则是把它们组成一组。

问： **我可以在<ol>或<ul>中添加其他文字或者元素吗？**

答： 不能，<ul>和<ol>元素的诞生只是为了配合<li>元素一起工作。

问： **那无序列表呢？我可以使要点与众不同吗？**

答： 可以，但先留着这个问题吧。以后我们讲到CSS和外观时会加以讨论的。

问： **如果我想在列表中添加列表呢？可以这么做吗？**

答： 当然可以。用<li>和<ol>或者<li>和<ul>组成内容，那么你的列表中就会带有列表（我们称为嵌套列表）。

```
<ol>
   <li>Charge Segway</li>
   <li>Pack for trip
       <ul>
          <li>cell phone</li>
          <li>iPod</li>
          <li>digital camera</li>
          <li>a protein bar</li>
       </ul>
  </li>
   <li>Call mom</li>
</ol>
```

嵌套列表

这里是<li>。它封入嵌套列表中。

问： **我想我已经基本理解块元素和内联元素了，但我对哪个元素可以在哪个元素中使用还觉得很混乱，或者，像你说的，什么可以"嵌套进"什么？**

答： 这是HTML中最难理解的要点之一。你在以后几章中也会学到这些，我们会从多个角度帮你理清思路。不过，我们先继续介绍嵌套。实际上，既然你提出了，我们接着就来讨论。

问： **HTML有有序列表和无序列表。还有其他列表类型吗？**

答： 确切来说还有一种类型：自定义列表。自定义列表的形式是：

列表中的每个项目都有一个项限<dt>和一个描述<dd>。

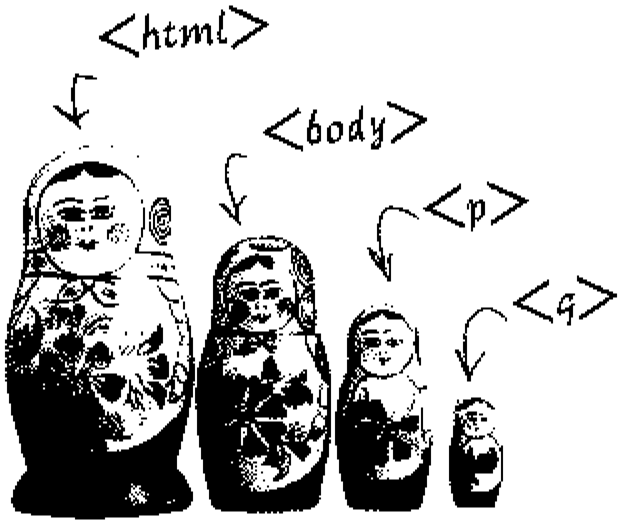
```
<dl>
   <dt>Burma Shave Signs</dt>
    <dd>Road signs common in the U.S. in the 1920s and 1930s advertising shaving products.</dd>
   <dt>Route 66</dt>
   <dd>Most famous road in the U.S. highway system.</dd>
</dl>
```

输入这些并测试。

问： **柏玛刮胡膏是什么？**

答： Burma Shave是20世纪早期的一家制造少泡沫刮胡膏的公司。他们1925年就开始在路边树立广告牌了，而且事实证明这些广告牌非常有效（看来有些人开车时分心了）。

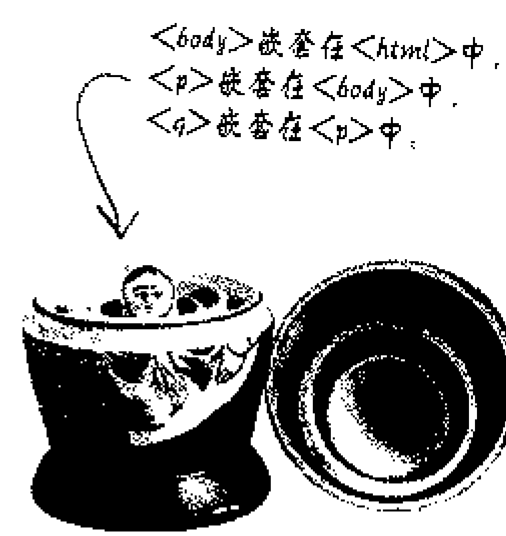
这些广告牌按组分成四、五或者六行，每行一句口号。一时之间，7000个这样的广告牌树立在美国的路边。现在，大多数都已经没有了，但有的地方还残留少许。



把一个元素放进另一个元素，叫做“嵌套”

当把一个元素放到另一个元素中，我们把这叫做嵌套。我们说，“**<p>**元素嵌套在**<body>**中”。你已经看到过许多元素嵌套在其他元素中的情况了。你把一个**<body>**元素放到一个**<html>**元素中，把一个**<p>**元素放到**<body>**中，把**<q>**元素放到**<p>**中等。你把**<head>**元素放到**<html>**中，把**<title>**放到**<head>**中。这就是HTML页面构建的方式。

HTML学得越深入,你就会越重视嵌套。别担心，不久你会很自然地用这种方式思考问题了。



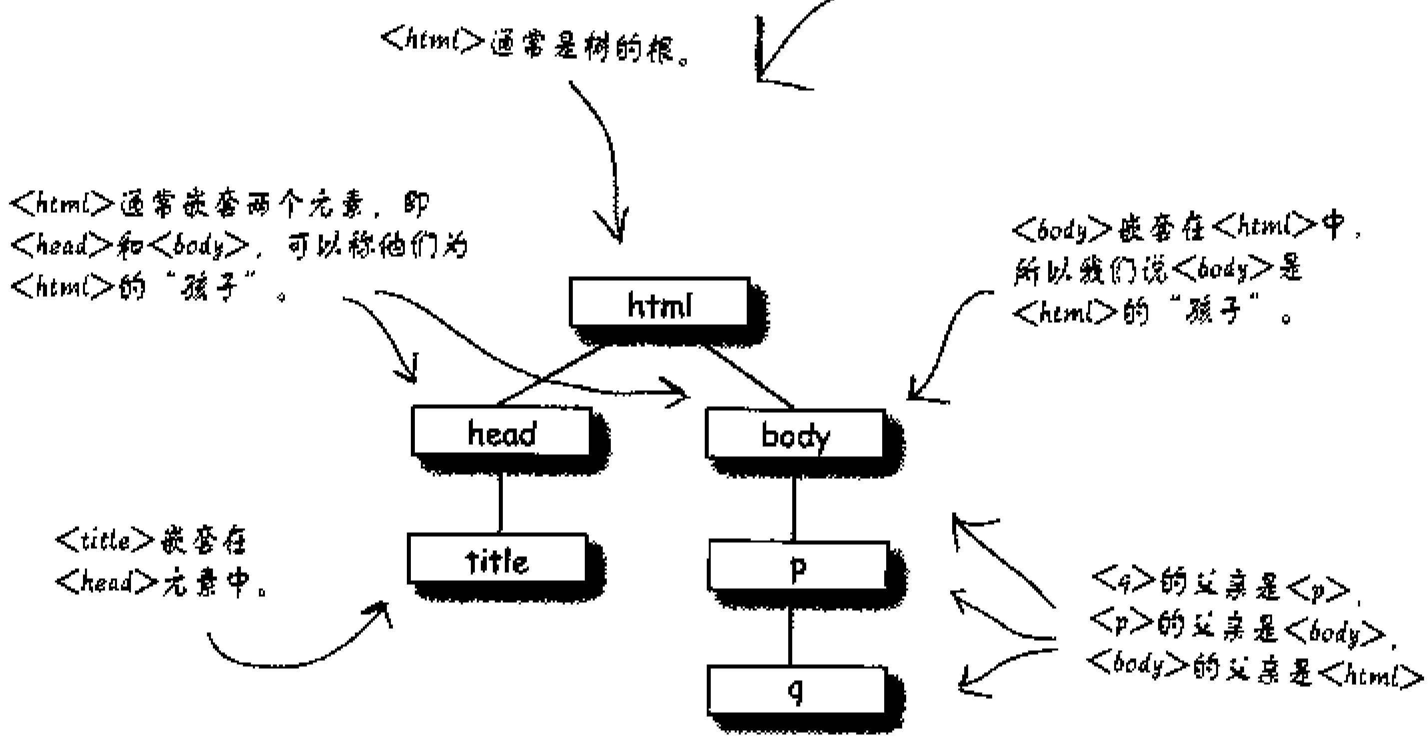
用图示来了解嵌套的关系

画网页中的嵌套元素就像画家族树。在顶部你会看到曾祖父，然后是他们的儿子和孙子。这里是个例子……

普通网页。

```
<html>
  <head>
    <title>Musings</title>
  </head>
  <body>
    <p>
      To quote Buckaroo,
      <q>The only reason
         for time is so
         that everything
         doesn't happen
         at once.</q>
    </p>
  </body>
</html>
```

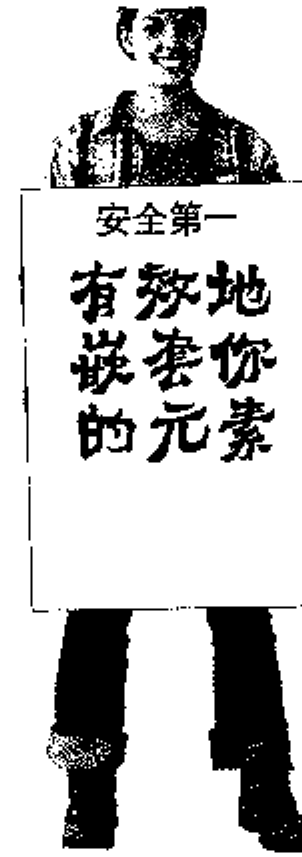
我们把它翻译成图表，每个元素做成一个格，元素间的连线代表嵌套。



使用嵌套来确保你的标记匹配

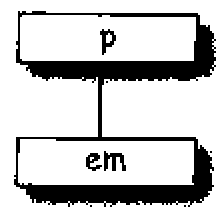
嵌套最重要的作用是，可以避免标记不匹配（以后会有更多回报的，耐心等待）。

什么是“标记不匹配”？它是怎么产生的？看看下面这个例子吧：




```
<p>I'm so going to blog <em>this</em></p>
```

这是<em>嵌套在<p>中。

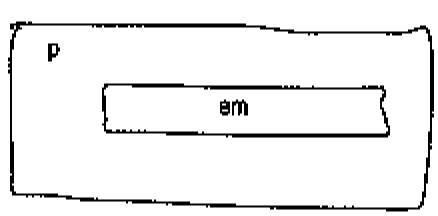


一切正常，但是在写某些HTML时很容易随手写错，比如：

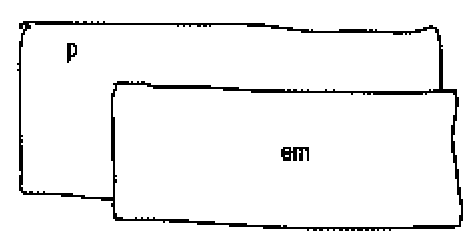
```
<p>I'm so going to blog <em>this</p></em>
```

错误：<p>标记在<em>标记前边结束了！<em>元素应嵌套在<p>里边。

假设你知道了嵌套，就知道**<em>**元素需要完全嵌套，或者说包含在**<p>**元素中。



正确：这里的<em>元素嵌套在<p>中。



错误：这里的<em>元素漏出了<p>元素，这就意味着嵌套错误。

结果会怎么样？

如果你喜欢俄罗斯轮盘机的话，可以把你的嵌套弄得混乱些。如果你写的HTML嵌套有误，网页可能无法在某些浏览器中使用。谨记嵌套，你可以避免标记不匹配，并确保你的网页在所有浏览器中都能顺利运行。这在我们的后续章节“工业化加强版HTML”中显得尤为重要。

扮演浏览器

以下是一个含有不匹配标记的HTML文件。你的任务是像浏览器一样指出全部错误。完成后，对照本章后的答案。



```
<html>
<head>
    <title>Top 100</title>
<body>
<h1>Top 100
<h2>Dark Side of the Moon</h2>
<h3>Pink Floyd</h3>
<p>
    There's no dark side of the moon; matter of fact <q>it's all dark.
</p></q>
<ul>
    <li>Speak to Me / Breathe</li>
    <li>On The Run</li>
    <li>Time</li>
    <li>The Great Gig in The Sky</li>
    <li>Money</li>
    <li>Us And Them</em>
    <li>Any Colour You Like</li>
    <li>Brain Damage</li>
    <li>Eclipse</li>
</ul>
</p>
<h2>XandY</h3>
<h3>Coldplay</h2>
<ol>
    <li>Square One
    <li>What If?
    <li>White Shadows
    <li>Fix You
    <li>Talk
    <li>XandY
    <li>Speed of Sound
    <li>A Message
    <li>Low
    <li>Hardest Part
    <li>Swallowed In The Sea
    <li>Twisted Logic
</ul>
</body>
</head>
```

我是谁?



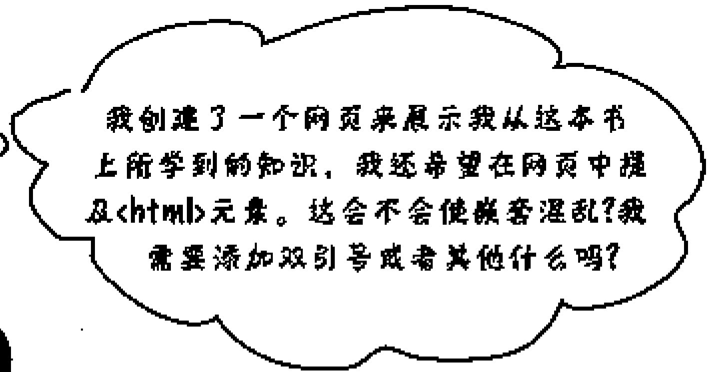
一群盛装的HTML元素，在玩着一个派对游戏——“我是谁”。他们会给你一个线索——而你则试着从他们的言语中猜出他们分别是谁。假设他们说的都是实话。辨别他们的身份，并填在右边的空格上。同时，写出每位出席者是块元素还是内联元素。

今晚出席者:

你所见过的所有光彩照人的HTML元素即将登场!

	名字	内联元素还是块元素?
我是1号标题。		
我准备链接到其他网页。		
我强调文段。		
我是列表,但我和顺序无关。		
我是个真正的换行者。		
我是个生活在列表中的项目。		
我会让自己的列表项目井井有条。		
我为图片而生。		
在段落中引用得靠我。		
用我来引用一段独立的文字。		





是的，那样做会出现问题。

原因是浏览器使用**<and>**来开始和结束标记，在HTML内容中使用它们会出问题。

不过，HTML为你提供了一个简单的缩写形式（字符实体）来指定这些特殊符号。这是他们工作的原理：被看作“特殊”的，或者你想在网页中使用但是可能在你的编辑器中无法输入(像版权标记)的字符，你只须查出缩写并输入到HTML。例如>符号的缩写是**>**，而<是**<**。

假如你想在你的网页中输入“The <html> element rocks.”，使用字符缩写，你可以这样输入：

```
The &lt;html&gt; element rocks.
```

你应该知道另外一个重要的特殊符号**&**。如果你想在HTML中使用**&**，可以使用字符实体**&**代替**&**本身。

那版权标记呢？其他所有标记和外文字母呢？你可以在这个URL中看到：

```
http://www.w3schools.com/tags/ref-entities.a2P
```

或者，你想要更详细的清单，用这个URL：

```
http://www.unicode.org/charts/
```

there are no Dumb Questions

问： 我一直不知道浏览器可以显示那么多不同的字符。在www.unicode.org网站上有数以亿计的不同字符和语言。

答： 小心。你的浏览器只会显示你的电脑或者设备安装的字体。也许你指望所有浏览器中都有www.w3schools.com上的基本实体，然而并不能保证你的浏览器一定可以全部显示。但是，如果你了解你的用户，需要在他们机器上顺利显示的外文字符就不是秘密了。

问： 你说&很是特殊，需要用实体&来显示它；在需要用&的地方要输入&来代替。但是要输入任何实体都要用到&，所以，就是说，对于>实体，我要输入>？

答： 不！说&是特殊的，准确来说是因为它是其他实体的首字符。使用实体名字时一定会用到&，而不是&的实体本身。记住输入实体的时候使用&，当需要在文章中用&的时候才用&来代替。

问： 浏览www.w3cschools.com时，我发现每个实体前边都有一个数字，它们的作用是什么？

答： 你可以在HTML中使用数字，像&#100，也可以使用实体名，它们的作用是相同的。但不是所有的实体都有名字，所以在这种情况下你只能使用数字。

破译位置密码的挑战

为了达到统治世界的愿望，Evel教授上传了一个私人网页来扩充羽翼。你已经收到部分拦截到的HTML片断，其中可能包含能找到他位置的线索。用你的HTML专业知识，破译代码并找到他的位置。这是他主页的部分文字：

```
There's going to be an evil henchman meetup
next month at my underground lair in
&#208;&epsilon;&tau;&#114;&ouml;&igrave;&tau;.
Come join us.
```

提示：浏览http://www.w3schools.com/tags/ref_entities.asp
或者输到HTML中，看看浏览器显示的结果。

元素大杂烩

<em>

使用这个元素来标识你想强调的文本。

<strong>

使用这个元素来标识你着重强调的部分。

<pre>

当你要浏览器显示你输入的文本时，使用这个元素来格式化文本。

当要建立链接时，你需要<a>元素。

<a>

<address>

这个元素告诉浏览器这段内容是个地址，例如你的联系信息。

一个用来换行的空元素……

……一个用来制作水平线(称为水平尺)的元素，比如开始一个没有标题的新部分。

<hr>

使用这个元素来进行短引用。比如，像"To be or not to be"或者"No matter where you go, there you are"。

<q>

需要显示一个列表：一个菜谱的成分，或者一个计划表；使用<ul>元素。

<p>

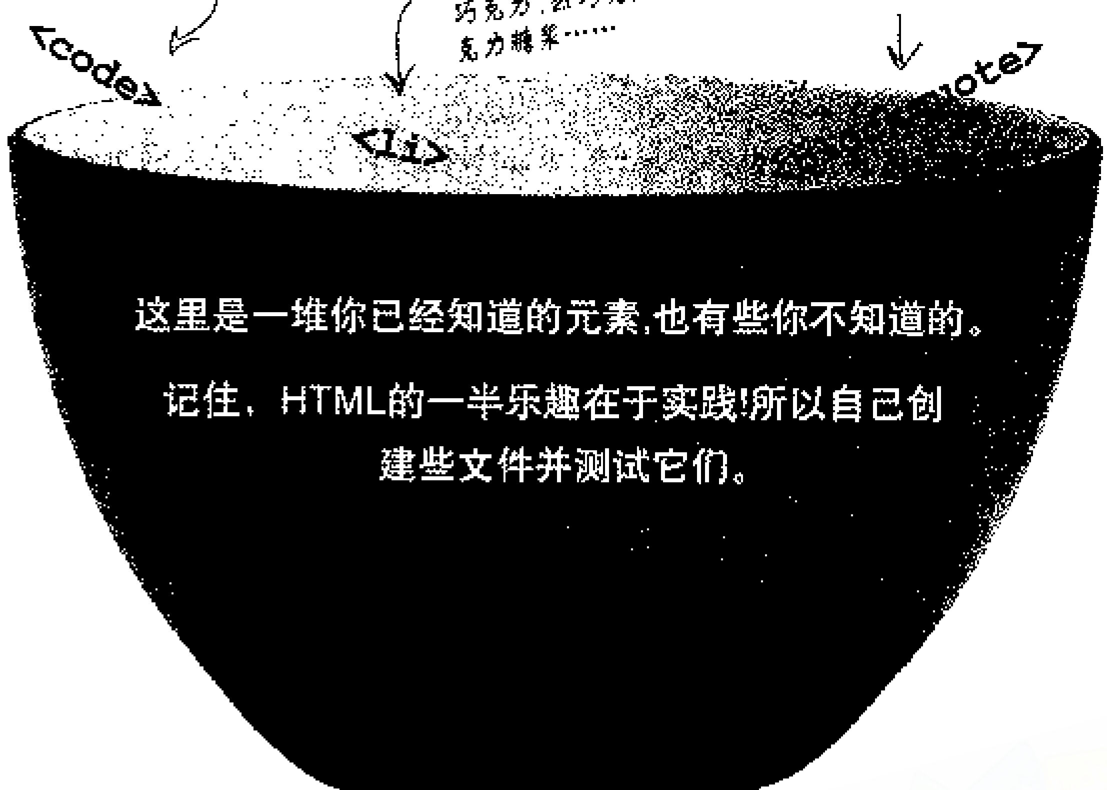
给我个段落，谢谢。

如果你需要一个有序列表，使用<ol>元素。

代码元素用来显示计算机程序的代码。

用于列表项目，例如巧克力，热巧克力，巧克力糖浆……

块引用用于长引用，比如强调书中的一大段文字。



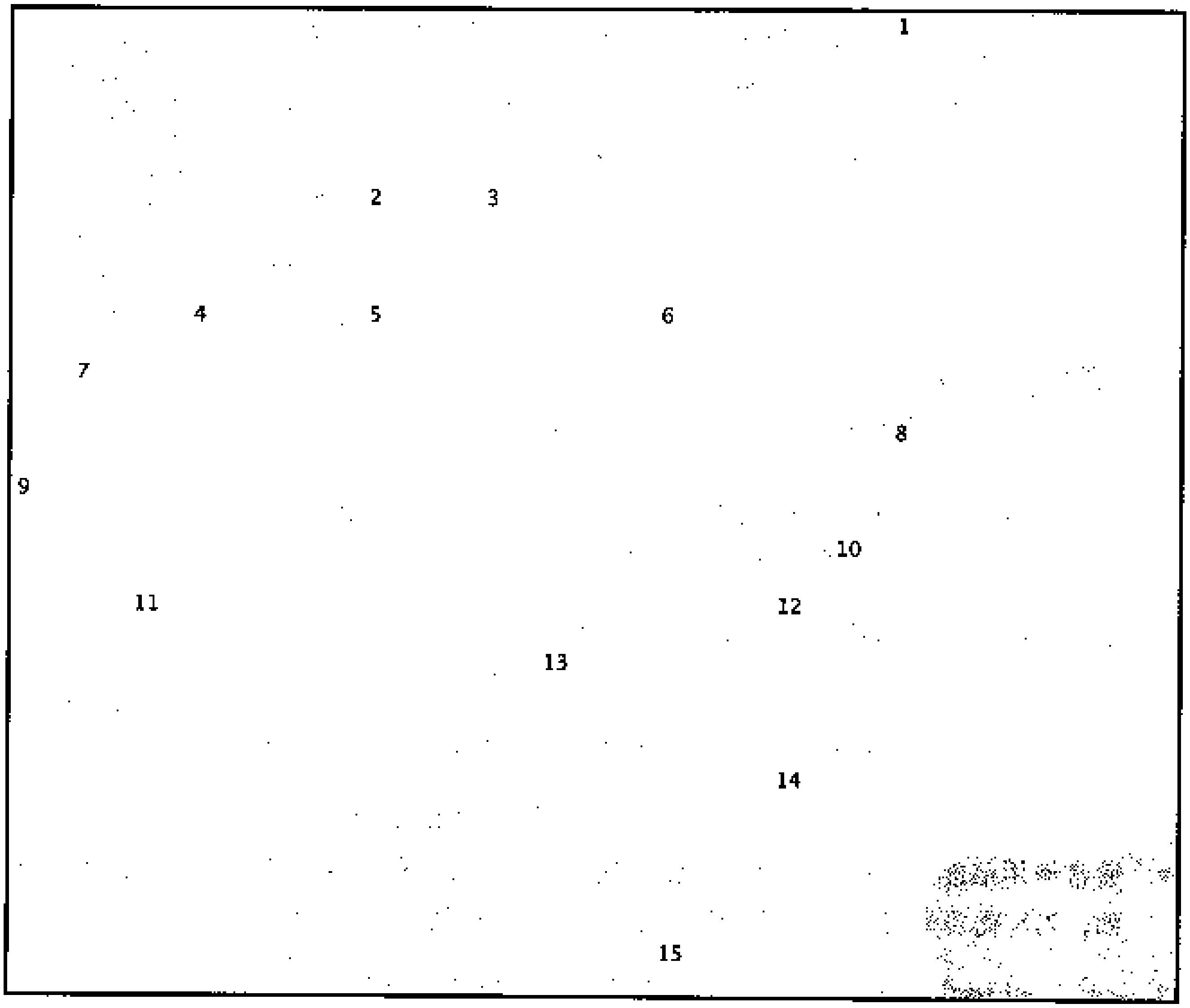


要点

- 在输入内容之前先计划好网页的结构。先画草图，再绘制略图，最后写HTML。
- 用大的块元素来创建你的网页，然后用内联元素修饰。
- 记住,无论什么时候,使用元素来告诉浏览器你的内容的含义。
- 通常使用最能匹配内容含义的元素。例如，当你需要列表时绝对不用段落表示。
- <p>、<blockquote>、<ol>、<ul>和<li>都是块元素。它们独立显示，文本前后有空行。
- <q>、<em>、<a>是内联元素。这些元素的内容和其他内容一起跟随在文字流中。
- 当你需要插入换行时，请使用
元素。
-
是个“空元素”。
- 空元素没有内容。
- 一个空元素仅由一个标记组成。
- 嵌套元素是一个完全包含在另外一个元素中的元素。如果你的元素嵌套正确,所有标记都会正确匹配。
- 结合使用两个元素创建HTML列表：使用<ol>和<li>创建有序列表，用<ul>和<li>创建无序列表。
- 当浏览器显示一个有序列表,它会给列表标上序号，无须你再费力了。
- 可以用start属性指定你自己的列表顺序。用value属性改变每个项的值。
- 可以在<li>中嵌入<ol>或者<ul>来建立嵌套列表。
- 在HTML中使用实体来显示特殊字符。

HTML填字游戏

该让你大脑右半球休息一下并让左半球开始工作了：所有词都是和HTML有关的，而且都是本章的内容。



横排提示：

2. 用于引用的块元素。
7. 网页的主要构建模块。
9. 需要两个元素。
10. 没有内容的元素。
11. <q>是这种类型的元素
13. 著名的易误导的路标。
14. Tony的交通工具。
15. 另外一个空元素。

竖排提示：

1. 驾车一起离开的人。
3. 用于这种列表。
4. 没有内容的空元素。
5. 把一个元素放到另外一个元素中的做法。
6. 用于这种类型的列表。
8. 电动滑板车的最高时速。
12. Tony不会做这些。

练习答案

这是用<q>元素改版Tony对老子的话的引用。

你测试自己的答案了吗？

这是改动过的部分……

```
<p>
  My first day of the trip!  I can't believe I finally got
  everything packed and ready to go.  Because I'm on a
  Segway, I wasn't able to bring a whole lot with me:
  cellphone, iPod, digital camera, and a protein bar. Just
  the essentials.  As Lao Tzu would have said, <q>A journey
  of a thousand miles begins with one Segway.</q>
</p>
```

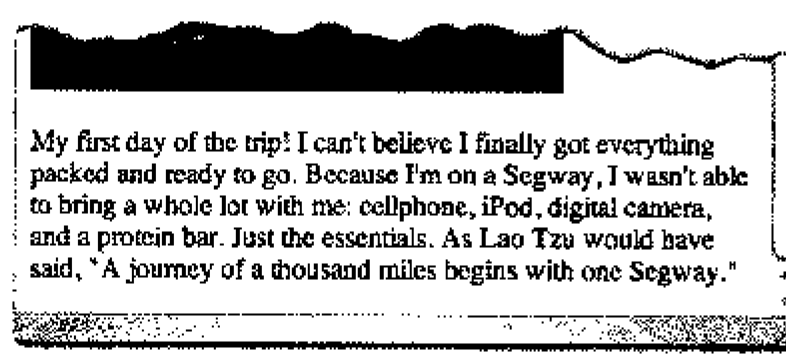
在引用开始前添加了<q>开始标记，在结尾添加</q>结束标记。

注意，我们也把双引号同时去掉了。

www.docin.com

这是测试结果……

OK，虽然看起来差别不大，但现在感觉好多了吧？



练习答案

这是Tony旅行日志中的另外一个列表：手提电话、iPOD、数码相机，还有蛋白棒。你可以在他7月14日的日志中找到。这是个无序列表。

在你的“journal.html”中做同样修改。看看和你预想的差多少？

```
<h2>June 2, 2005</h2>
<img src="segway1.jpg">
<p>
  My first day of the trip!  I can't believe I finally got
  everything packed and ready to go.  Because I'm on a Segway,
  I wasn't able to bring a whole lot with me:
</p>
<ul>
      <li>cell phone</li>
      <li>iPod </li>
      <li>digital camera</li>
      <li>and a protein bar</li>
</ul>
<p>
  Just the essentials.  As
  Lao Tzu would have said, <q>A journey of a
  thousand miles begins with one Segway.</q>
</p>
```

首先结束前边的段落。

开始无序列表。

每个项目都放到<li>元素中。

结束无序列表。

我们需要开始一段新的段落。



扮演浏览器 答案

```
<html>
<head>
    <title>Top 100</title>
<body>
<h1>Top 100
<h2>Dark Side of the Moon</h2>
<h3>Pink Floyd</h3>
<p>
    There's no dark side of the moon; matter of fact <q>it's all dark.
</p></q>
<ul>
    <li>Speak to Me / Breathe</li>
    <li>On The Run</li>
    <li>Time</li>
    <li>The Great Gig in The Sky</li>
    <li>Money</li>
    <li>Us And Them</em>
    <li>Any Colour You Like</li>
    <li>Brain Damage</li>
    <li>Eclipse</li>
</ul>
</p>
<h2>XandY</h3>
<h3>Coldplay</h2>
<ol>
    <li>Square One
    <li>What If?
    <li>White Shadows
    <li>Fix You
    <li>Talk
    <li>XandY
    <li>Speed of Sound
    <li>A Message
    <li>Low
    <li>Hardest Part
    <li>Swallowed In The Sea
    <li>Twisted Logic
</ul>
</body>
</head>
```

- 缺少</head>结束标记。（指向 <title>Top 100</title> 之后）
- 缺少</h1>结束标记。（指向 <h1>Top 100）
- <p>和<q>错误嵌套；</p>标记在</q>之后。
- 我们在需要
结束标记的地方放置了结束标记。
- 这里的</p>结束标记没有任何一个<p>开始标记与之匹配。
- 这些标题的</h2>和</h3>结束标记混淆了。
- 开始了一个
列表，但它对应
结束标记。
- 缺少所有的
结束标记。
- 这与开始的<ol>开始标记不匹配。
- 这里缺少</head>标记。我们还缺少个</html>标记。

练习答案

我是谁？



许多盛装的HTML元素，在玩着一个派对游戏——“我是谁？”。他们会给你一个线索——而你则试着从他们的言语中猜出他们分别是谁。假设他们说的都是实话。辨别他们的身份，并填在右边的空格上。同时，写出每位出席者是块元素还是内联元素。

今晚出席者：

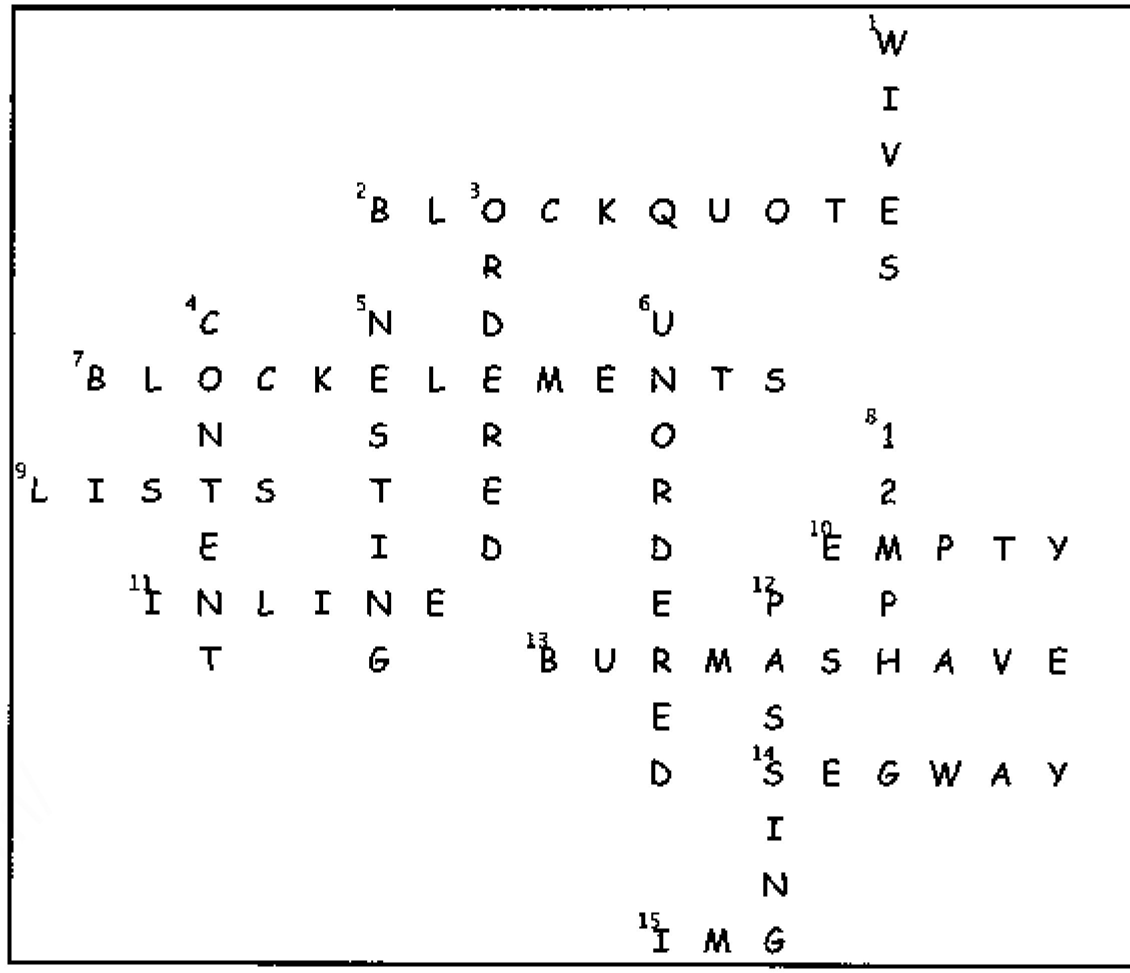
你所见过的所有光彩照人的HTML元素即将登场!

	名字	内联元素还是块元素？
我是1号标题。	h1	block
我准备链接到其他网页。	a	inline
我强调文段。	em	inline
我是列表,但我和顺序无关。	ul	block
我是个真正的换行者。	br	
我是个生活在列表中的项目。	li	block
我会让自己的列表项目井井有条。	ol	block
我为图片而生。	img	inline
在段落中引用得靠我。	q	inline
用我来引用一段独立的文字。	blockquote	block

空白的，
既不是内联元素，也不是块元素。它创建了换行，但是不像块元素那样在周围显示空白。

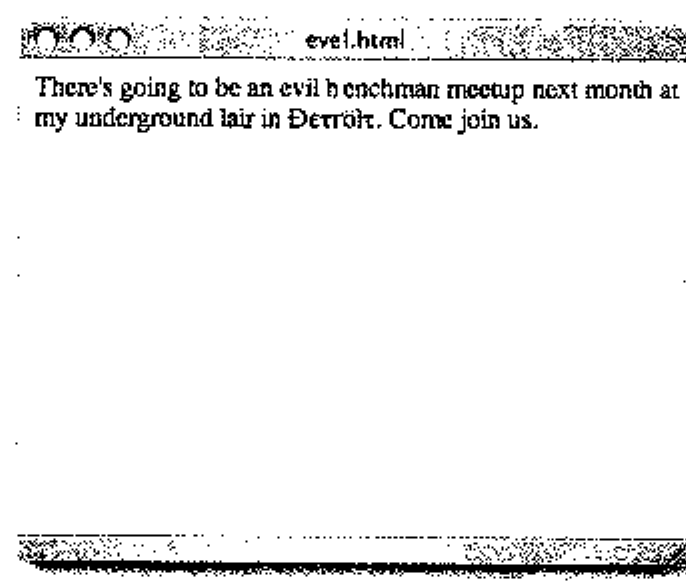
对此，我们还没有做详细的介绍。没错，<img>是内联元素。先记住这个，其他的我们将在第5章中介绍。

练习答案



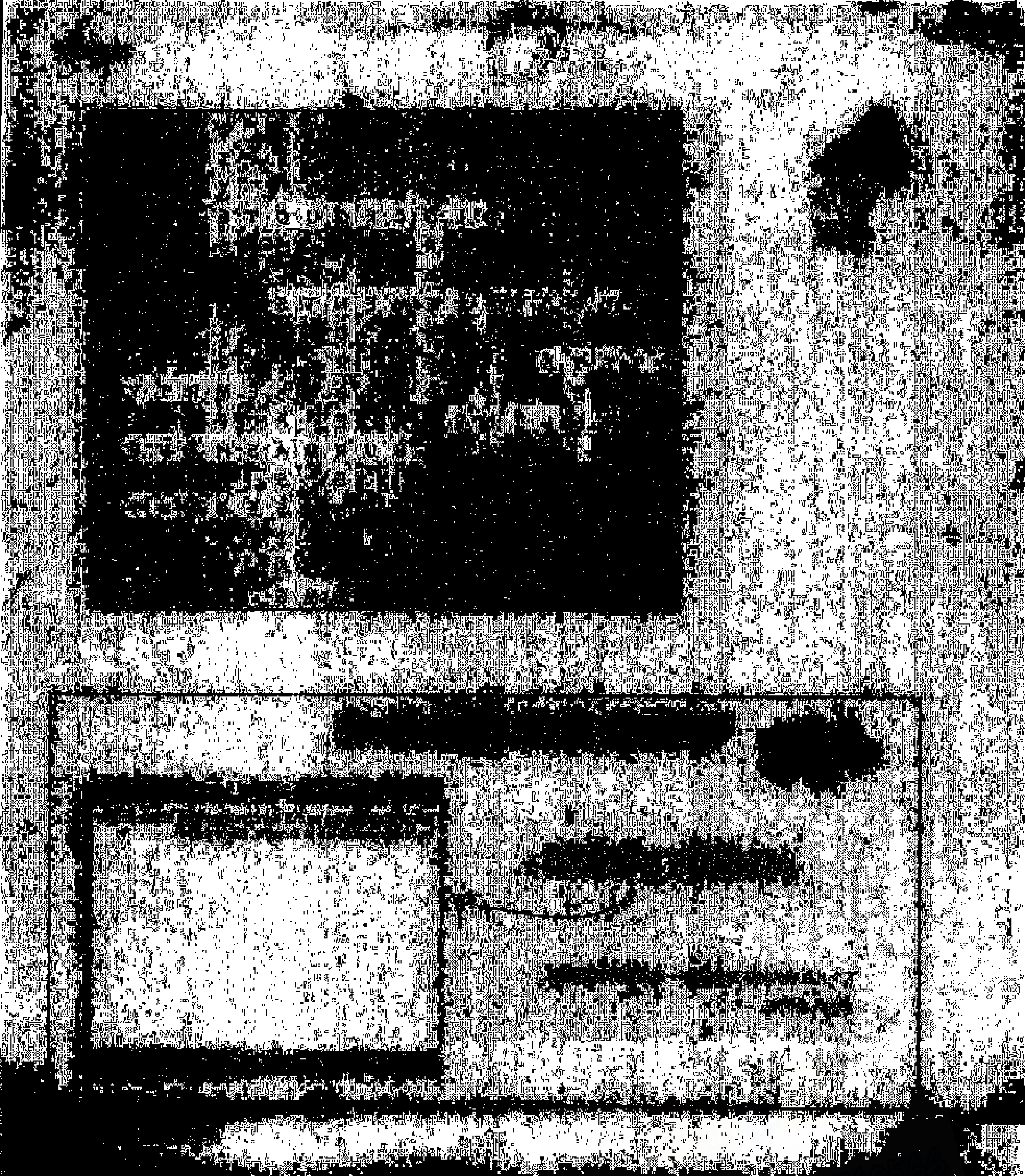
破译位置密码的挑战

你可以查找每个实体，或者输入它们。
无论采用哪种方式，答案应该是底特律！



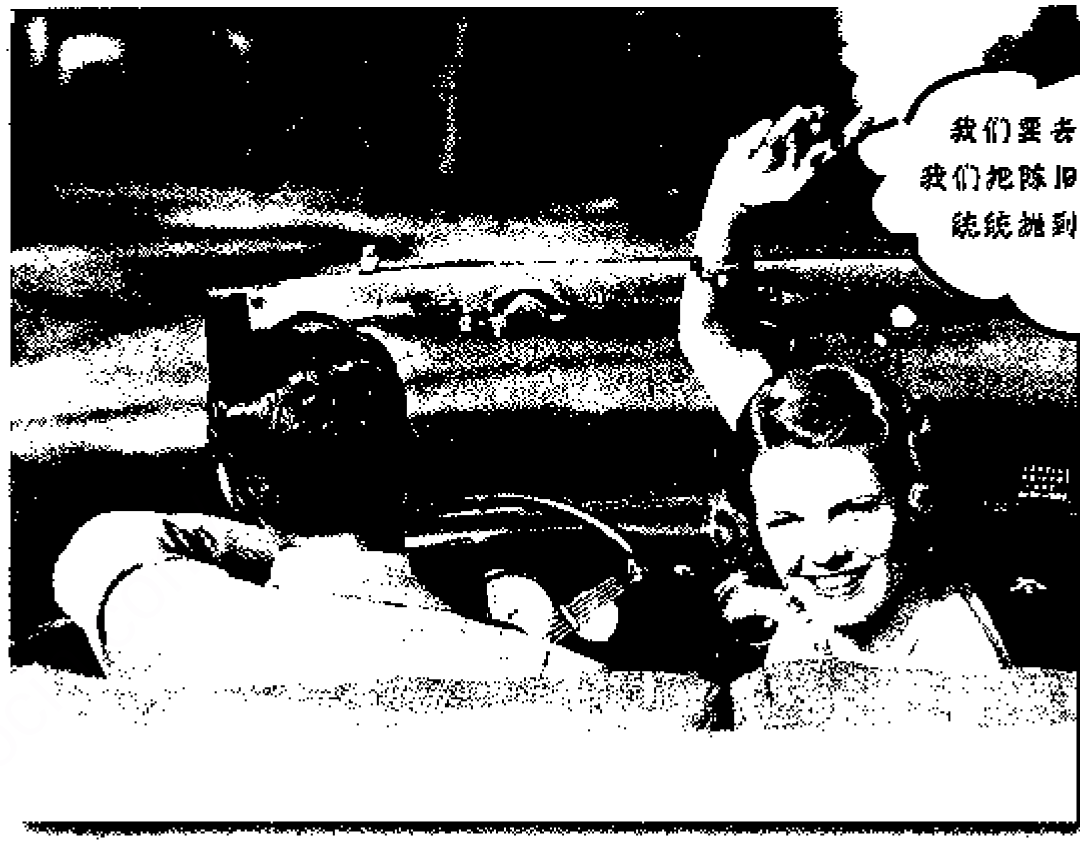
下个月在我的地下室将会有一个邪恶的党羽集会。

来加入我们吧。



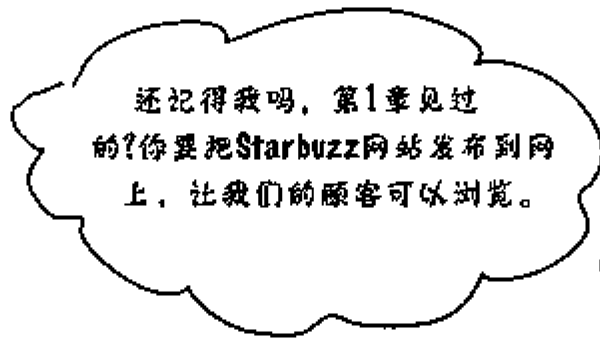
4 开始链接

Web镇之旅



网页是互联网给我们的最好的礼物。迄今为止，你已经学会了创建只能在自己电脑里生存的HTML页，还只能链接到你电脑中的其他页。我们将要来个翻天覆地的改变。在这一章我们鼓励你把这些网页搬到网上去，你的朋友、fans和顾客都可以看到。我们同样会通过破译代码h，t，t，p，：，/，/，w、w、w等揭示链接到其他网页的秘密。那么，收拾好你的行李，下一站就是Web镇。

提醒：一旦你到了Web镇，就可能永远回不来了。给我们寄明信片哦！





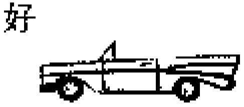
发布Starbuzz(或者你自己)的主页到Web上

你就要发布Starbuzz网站了。当然，如果是发布你自己的网站，那就更棒了。这个过程可能比你想象的还要快。你所要做的就是找到一个“Web主机代理商(Web Hosting Company)”(以后我们就叫主机代理商)，来让他们的服务器为你的页面服务，把你的网页从你的电脑中拷贝到他们的服务器上。

当然这有助于你理解从本地文件夹到服务器文件夹的一一对应过程，还有一旦把页面放到服务器上后，又该如何告知浏览器。我们会把这些都教给你。现在，我们来谈谈怎样把这个主页发布到网上去。这是你要做的：

1. **找一个主机代理商。**
2. **选择网站的名字(例如“www.starbuzzcoffee.com”)。**
3. **寻找把文件从电脑传到主机的途径(有许多途径)。**
4. **把新网站告诉你的朋友、家人和fans，给他们个惊喜。**

即使你此刻还没有做好建网站的准备，只要跟上我们的步骤，就能学到许多日后有用的东西，所以，让我们准备好从HTML这来个急转弯吧……


Web 浏览

寻找主机代理商

为了将网页发布到Web上，你需要一个全日工作的网络服务器。最好的方法是找到一家主机代理商，至于如何让服务器一直工作，这些细节就让他们来考虑吧。不要担心，找一家主机代理商还是相当方便和廉价的。

你想问选择哪家公司?我们也希望你和Head First Hip Web Hosting公司签约，但是这个公司实际上并不存在。所以，你必须自己做些功课了。找一个可以上传你网页的主机应该不是很难，就像选择有线电视公司一样，有许多选择和方案。你需要货比三家，在价钱和服务之间找到平衡。

有个好消息：你可以从囊空如洗的状态开始。如果你需要一些附加，还可以升级。我们不能推荐某个特定的代理商，但可以告诉你一些可供选择的信息。我们在这里也列出了一些比较受欢迎的代理商：

http://www.headfirstlabs.com/providers.html

交易须知：如果某个主机代理商很有实力，我们可以选择他。

放轻松

学习这本书并不一定要把网页发布到Web上。

如果你的网页发布到Web上的话，会有更多乐趣，不过也可以在自己的电脑上完成书中的剩余部分。

无论是哪种情况，跟着学完后面几页，你就会明白网站发布的全过程。

选择主机快速指南

我们不能历数选择主机代理商的方法（毕竟，这本书是介绍HTML和CSS的），但是我们还是会在正确的方向上推你一把。这是你选购时需要考虑的特征：

- 技术支持：这家公司能否提供一个能处理技术问题的优秀系统?优秀的公司会用电话或者E-mail回答你的问题。
- 数据传送：这是衡量你的网页和数据在特定时间内能提供给浏览者的数量的标准。大多数主机代理商以为小网站提供合理的数据传送量为基本目的。如果你希望自己的网站客如云来的话，在这点上就要好好留意了。
- 备份：你的主机代理商会定期给你的网页和数据做备份，以便当服务器出现硬件故障时能恢复吗?
- 域名：你的主机代理商出售的时候包含了域名吗?下一页我们会详细讨论这个问题。
- 可靠性：大多数主机代理商声称他们的网站正常工作时间保持在99%以上。
- 赠品：附送的赠品是否还包括E-mail地址、论坛或者对脚本语言的支持等额外服务（有些在以后会有用）。

你好，我的名字（域名）是……

即使还没听说过域名，你也已经看过和使用过上亿次了。你知道的：google.com，yahoo.com，amazon.com，Disney.com，或许还有一些你不想我们提及的。

那什么是域名呢?那是个用来定位你的网站的独一无二的名字。下面是个例子：

这部分是域名。

www.starbuzzcoffee.com

这部分是域中的特定服务器名。

不同用途的域的结尾也不同：.com，.org，.gov，.edu，同样还有不同国家的域：.co.uk，.co.jp之类。选择域的时候要挑最适合你的那个。

有许多原因使你必须重视域名。如果想让自己的网站有一个独一无二的名字，就要有自己的域名。域名也用于把你的网页链接到其他网站(我们以后会谈这个问题)。

还有一件事你应该知道。域名由一家权威机构（称为ICANN）控制，来确保每个人一次只能使用一个域名。同时，你必须每年交小量的注册资金来保持域名的使用权。



如何获得一个域名？

简单的答案是让你的主机代理商来考虑这个问题。他们经常会把域名注册放到他们的系列服务中去。然而，还有上百家公司会乐意帮忙的——你可以在这里找到他们的列表。

`http://www.internic.net/regist.html`

找到主机代理商后，你可能要自己去寻找和注册自己的域名了。你会发现找你的主机代理商应该是最简单的方法。

there are no Dumb Questions

问： **为什么称其为“域名”而不是“网站名”？**

答： 因为它们是不同的东西。如果你看到www.starbuzzcoffee.com，那是一个网站名，但是只有“starbuzzcoffee.com”的话就是域名。你可以用同一个域名来创建其他网站，像corporate.starbuzzcoffee.com或者employees.starbuzzcoffee.com。所以说域名是可以用在许多网站上的。

问： **如果我准备为Starbuzz获得一个域名，是不是应该去要个像www.starbuzzcoffee.com这样的名字？看起来似乎每个人的网站都是以www开头的。**

答： 再次提醒你，不要混淆域名和网站名：starbuzzcoffee.com是个域名，而www.starbuzzcoffee.com是个网站名。购买域名就像购买一块土地，比如100mainstreet.com，在这块土地上你可以随你喜欢来决定建筑的数量，举例来说，你可以建：home.100mainstreet.com，toolshed.100mainstreet.com和outhouse.100mainstreet.com。所以www.starbuzzcoffee.com只是starbuzzcoffee.com域名中的一个网站。

问： **域名很重要吗？我真的需要一个？我的主机代理商说我使用他们给的名字www.dirtcheaphosting.com就可以了？**

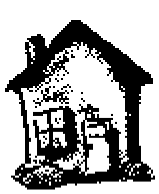
答： 如果能满足你的需要，使用他们的名字没有任何问题。但是，这里还有个缺点：如果你想选择另外一家主机代理商或者他们歇业了，知道你的网站的人就不太容易再找到它了。换一个角度来说，如果你有域名，你就可以把它更新到新的主机代理商了(你的用户甚至都不会发觉你的更改)。

问： **如果域名是独一无二的，那意味着有可能某人已经在使用我想要的那个域名了。我怎么得知呢？**

答： 问得好！大多数提供域名注册服务的公司都会帮你查看你想要的名字是不是已经存在(就像搜索空执照号码一样)。你可以在这里找到这种公司的列表：

http://www.internic.net/regist.html

这是一个需要独立思考的练习。我们都希望能给你帮助，但是这次你只能去问本书作者了(他可能去巴厘岛度假了)。



在家里试试这个

该为你的网站找一家主机代理商并获得一个域名了。记住，你可以浏览“Head First labs”去寻求意见或者资源。同样也请记住，你不用做这个也能学完这本书。

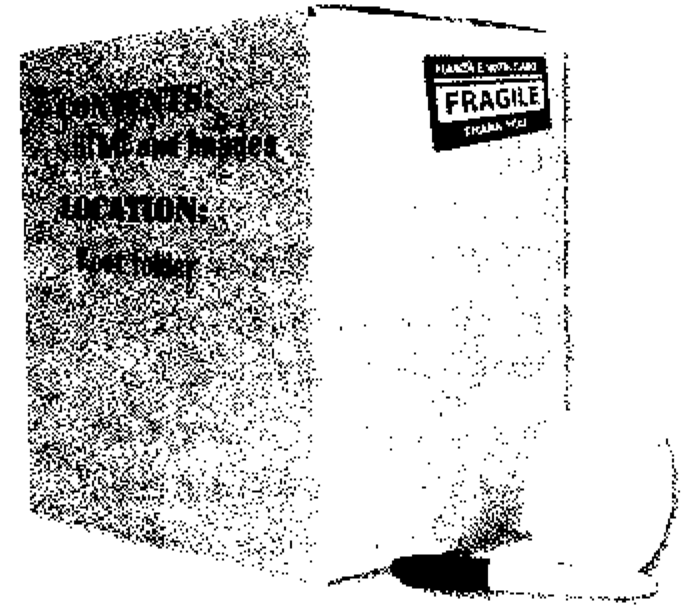
My Web Hosting Company: ______________________

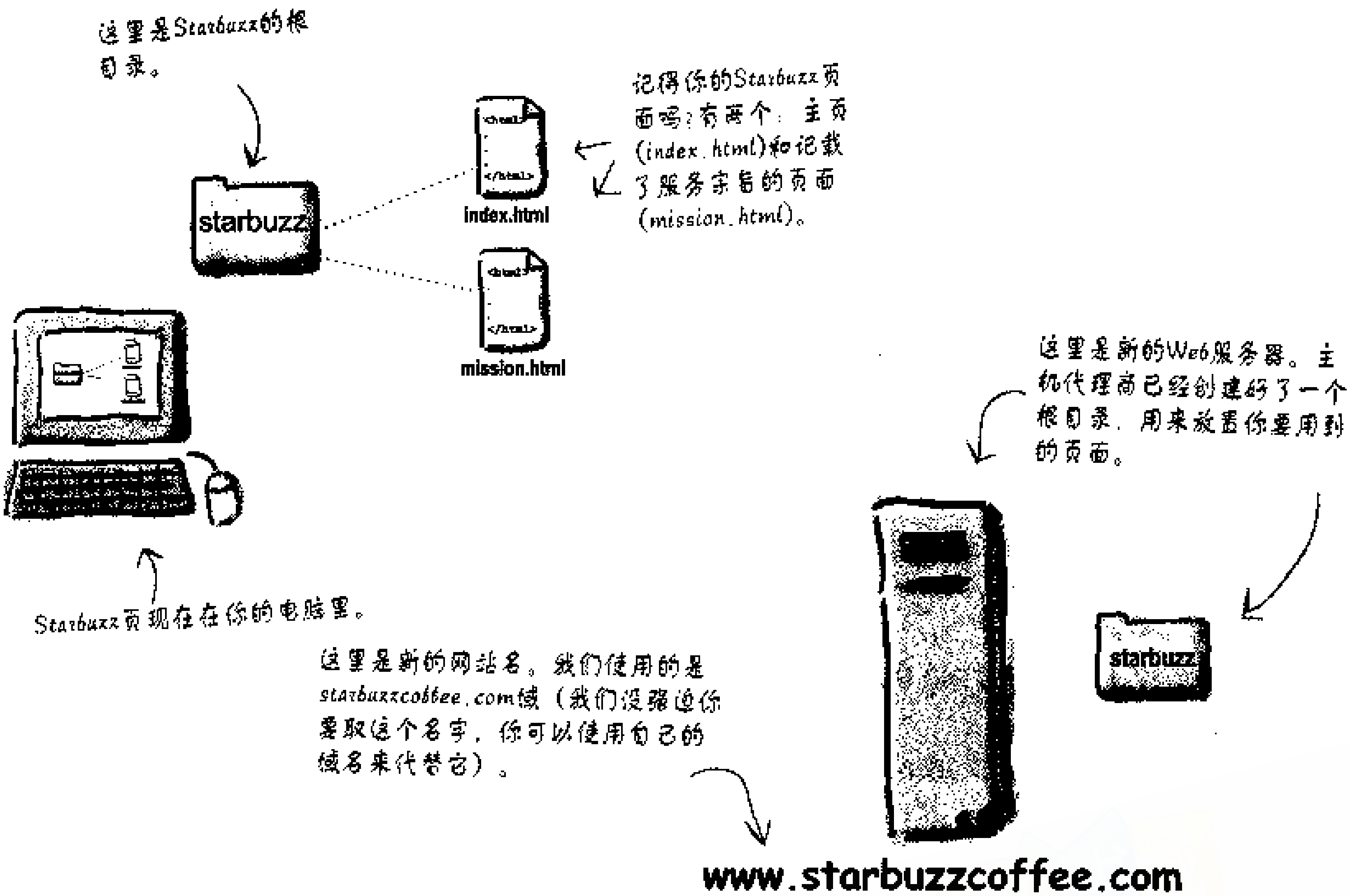
My Domain Name: ______________________

搬迁

恭喜你!你找到了主机代理商，获得了域名，这样就为你的网页准备好了服务器（如果你还没有这么做，继续认真学习以下内容，因为这个很重要）。

现在该怎么办?嗯，当然是先搬进来。所以，摘下“出售”的牌子并收集好全部文件，我们准备把它们移到新的服务器上去。像普通的搬家一样，目的是把东西从旧家搬到新家。在Web上，我们考虑的是从你的根目录到Web服务器的根目录。让我们回到Starbuzz，一步一步来。这是目前的情况：





there are no Dumb Questions

问： 等一下，再问一次，什么是"根目录"？

答： 网页的最顶层目录就是根目录。在Web服务器中，因为根目录中的东西有可能在Web上被访问，所以根目录显得尤为重要。

问： 我的主机代理商为我的根目录起名为"mydomain.com"。这会有什么问题吗？

答： 没关系。主机代理商用许多名字来称呼根目录。重要的是你知道你的根目录在服务器的哪个位置，这样你才可以把文件传过去(我们会马上教你)。

问： 还得问一下。我们已经把所有的网页放到了一个称为根目录的文件夹中。现在是要把这些都复制到服务器的根目录中去吗？

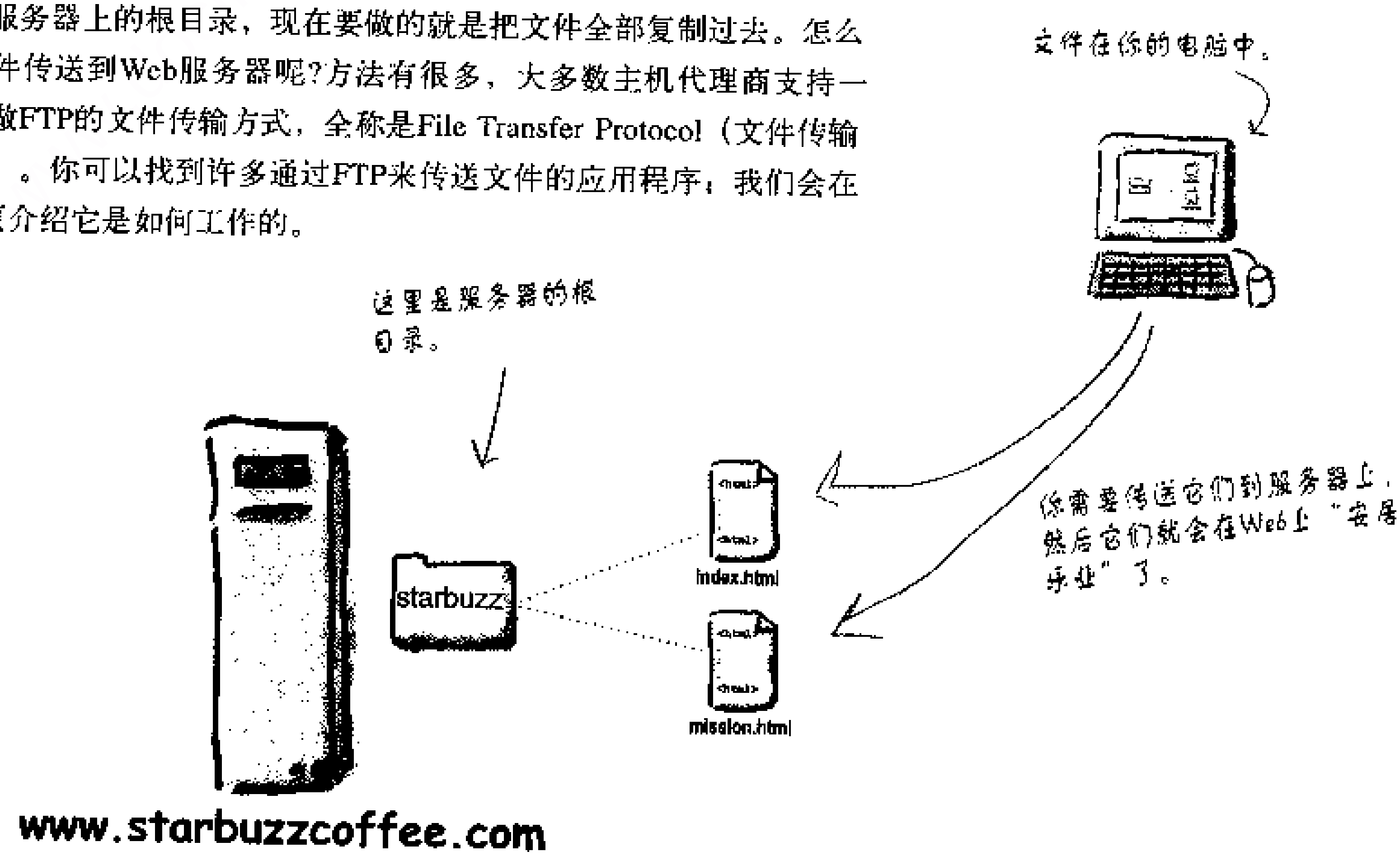
答： 准确来说是的。你要把所有的网页从你的电脑中拷贝到主机代理商提供给你的服务器的根目录下。

问： 那子目录怎么办？比如images文件夹，也要把它复制过去吗？

答： 对，就是要把你电脑里根目录下的所有文件、页面、还有子目录复制到服务器那里。所以如果在你电脑中有一个images文件夹，那么服务器中也应该有一个。

把你的文件复制到根目录

你现在距把Starbuzz Coffee发布到Web上只有一步之遥了：你已经确定了服务器上的根目录，现在要做的就是把文件全部复制过去。怎么把文件传送到Web服务器呢？方法有很多，大多数主机代理商支持一种叫做FTP的文件传输方式，全称是File Transfer Protocol（文件传输协议）。你可以找到许多通过FTP来传送文件的应用程序，我们会在下一页介绍它是如何工作的。



尽量在两页中介绍你可能用到的FTP

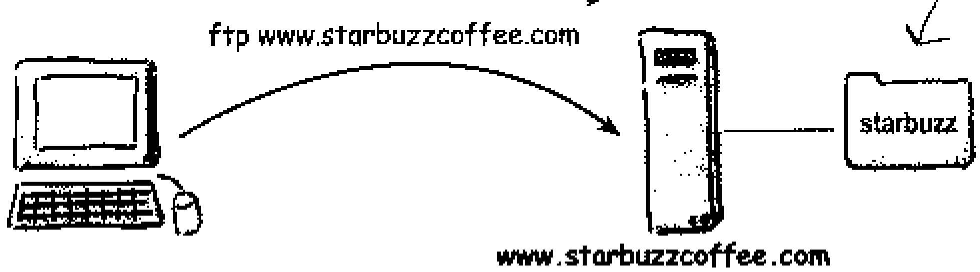
严格地说，这真正是本关于HTML和CSS的书，但是我们不想把你留在河中却不给你桨。所以，这里简单介绍一下如何使用FTP把文件传送到Web上。谨记你的主机代理商可能会有关于如何把文件传送到他们的服务器上的建议(你是付了钱的，就让他们来教你吧)。在这几页之后，我们会离开岔路回到HTML和CSS，一直到本书结束(我们保证)。

假设你已经有了一个FTP应用软件。有些是命令行驱动的，有些有完全图形化的界面，还有些被做成了像Dreamweaver和GoLive这样的软件。它们都有相同的命令，有些你需要自己键入命令，而有些你只需点击图形命令就可以了。以下是FTP的使用步骤：

❶ **首先用FTP连接到你的服务器。**

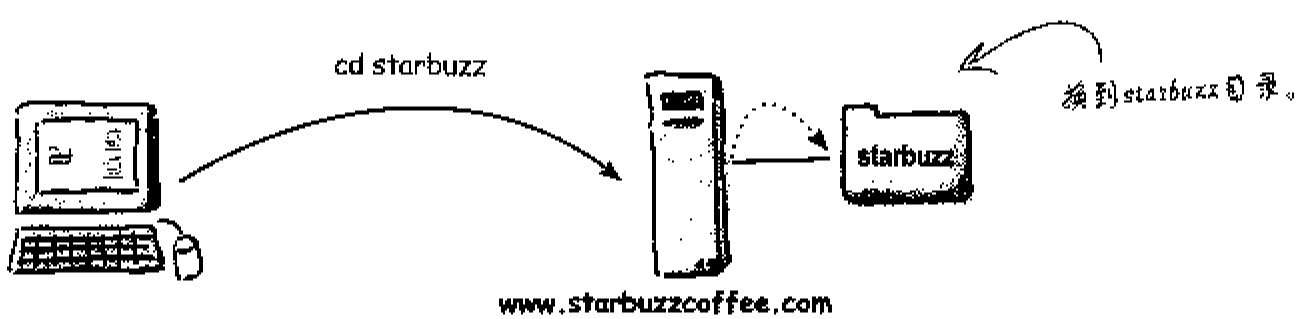
你需要有主机代理商提供的用户名和密码来连接。

"文件夹"(folder)和"目录"(directory)这两个词是对等的。大多数FTP应用程序用"目录"。



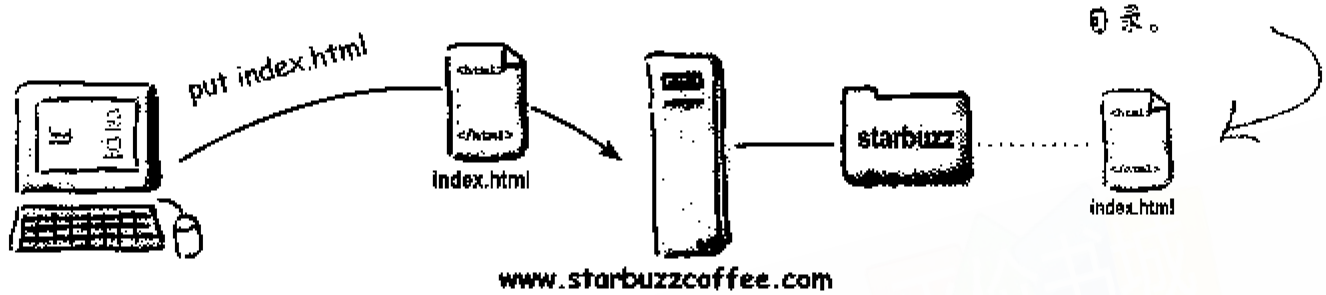
就是说，在传输文件前，确保你在服务器中的文件夹"starbuzz"中。

❷ **使用"cd"命令来从当前目录进入到你想传送其中文件的目录。**



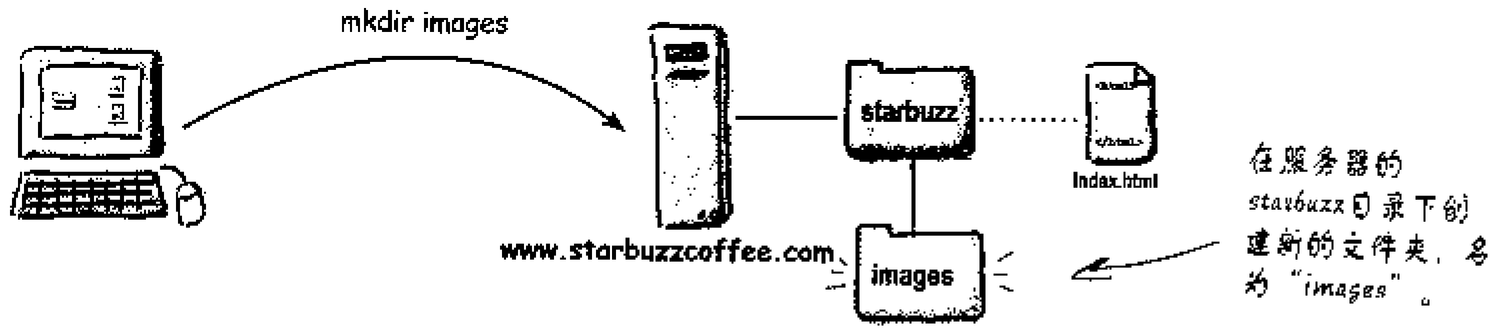
❸ **用"put"命令把文件传送到服务器。**

传送"index.html"的副本到服务器的当前目录。

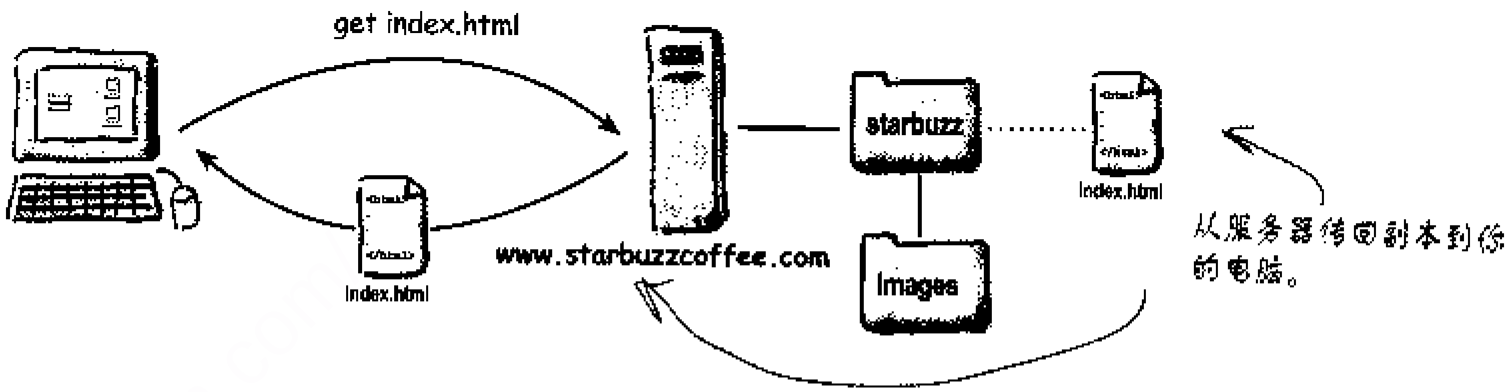


4 也可以用“mkdir”命令在服务器上创建一个新的目录。

就像建立一个新文件夹一样，不过不是在你电脑上而是在服务器上。



5 你可以用“get”命令返回文件。



让我们总结一下。这里是使用命令行驱动的FTP的例子：

大多数FTP应用程序都带有友好的图形界面，所以如果你有这样的应用程序，就可以跳过这里。

```
File Edit Window Help Jam
%ftp www.starbuzzcoffee.com
Connected to www.starbuzzcoffee.com
Name: headfirst
Password:******
230 User headfirst logged in.
ftp> dir
drwx------  4096 Sep  5 15:07 starbuzz
ftp> cd starbuzz
CWD command successful
ftp> put index.html
Transfer complete.
ftp> dir
-rw-------  1022 Sep  5 15:07 index.html
ftp> mkdir images
Directory successfully created
ftp> cd images
CWD command successful
ftp> bye
```

连接并登录。

获得当前目录。

一个叫做starbuzz的目录。

进入starbuzz目录。

传送index.html。

看看这个目录，有index.html。

为图片创建一个目录，并用bye命令退出。

FTP命令

无论是命令行驱动的FTP应用程序还是使用图形界面的FTP应用程序，命令或者操作都是相似的。

- dir：获得当前目录的文件清单。
- cd：进入另外一个目录。“..”也代表上溯一个目录。
- pwd：显示当前你所在的目录。
- put<filename>：传送指定文件到服务器。
- get<filename>：从服务器接收指定的文件到你的电脑。

there are no Dumb Questions

问： **我的主机代理商让我使用SFTP而不是FTP。它们之间有什么不同？**

答： SFTP（Secure File Transfer Protocol，安全文件传输协议）是FTP的更安全版本，工作原理基本相同。在购买前应确保你的FTP应用程序支持SFTP。

问： **想升级网站时，是不是要先在自己的电脑里编辑文件，然后再把它们传送到服务器上？**

答： 是的，对小型网站来说，这是普遍的做法。在把文件传送到服务器之前，用你的电脑测试你的更改并确保它是按你想要的方式正常运行的。对大型网站来说，组织者经常会创建一个测试站和一个实况站，这样他们可以先在测试站预览修改，然后再移到实况站。

如果你用类似于Dreamweaver或GoLive这样的工具，可以用它们在自己的电脑上测试你的改动，文件保存后会被自动传送到网站。

问： **我可以直接在Web服务器上修改我的文件吗？**

答： 这通常不是一个好主意，因为浏览者可能会在你预览和修复前看到你网页的改变和错误。

也就是说，某些主机代理商会允许你登录服务器并在服务器上修改。你通常需要使用DOS或者Linux的DOS命令提示，这取决于服务器上运行的是哪个操作系统。

流行的FTP应用程序

这里是一些在Mac和Windows下比较常用的FTP应用程序：

对于Mac OS X系统：

- Fetch（http://fetchsoftworks.com/）是一个在Mac下最常用的FTP应用程序。收费
- Transmit（http://www.panic.com/transmit）。收费
- Cyberduck（http://cyberduck.ch/）。免费

对于Windows系统：

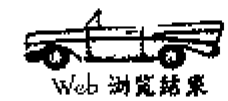
- Smart FTP（http://www.smartftp.com/download/）。收费
- WS_FTP（http://www.ipswitch.com/products/file-transfer.asp）。测试版本免费，专业版收费

大多数FTP应用程序都有测试版本，购买前你可以先下载试用。

在家里试试这个

这是另外一个家庭作业（每做完一项在前面方框内打勾）：

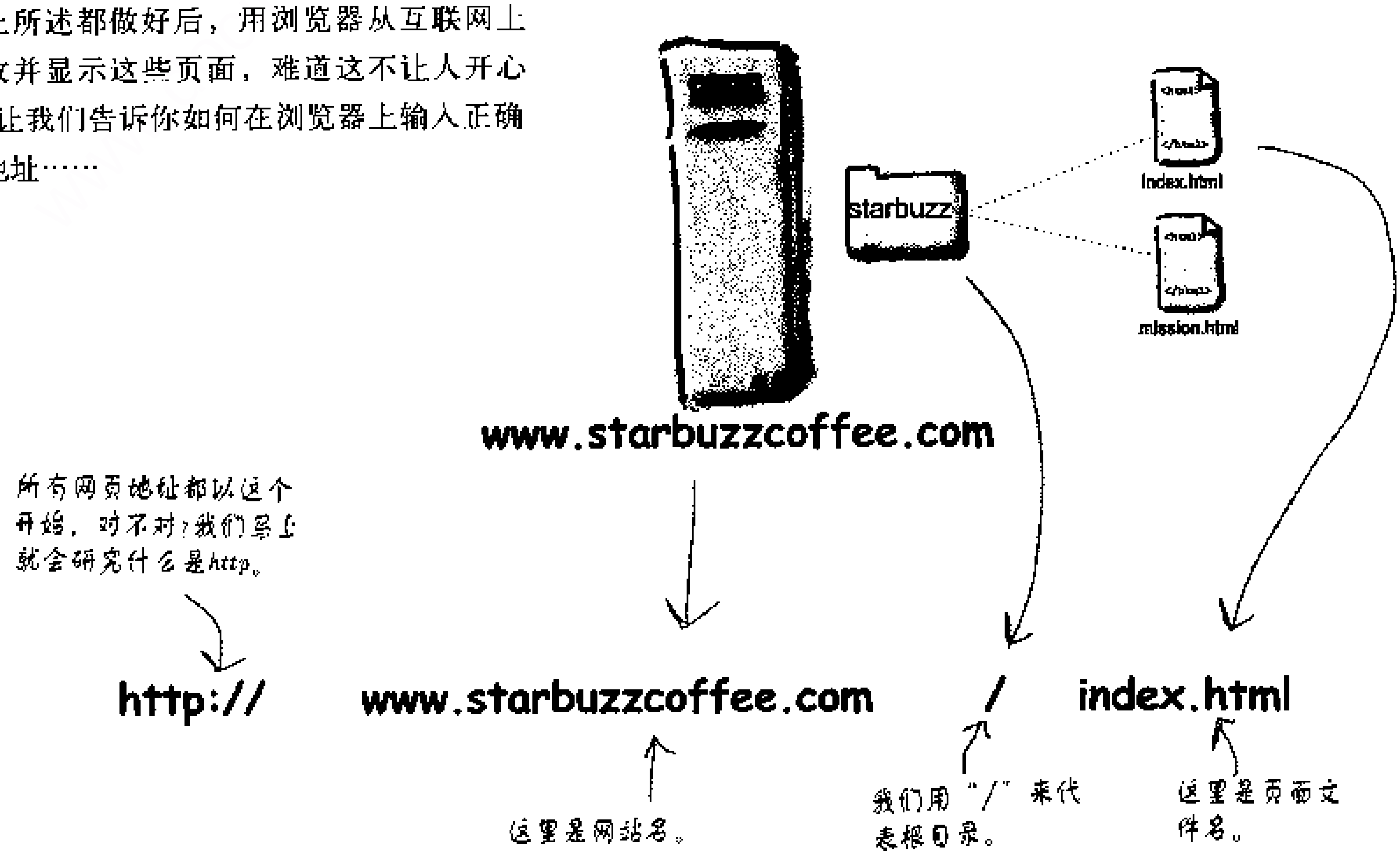
- ❑ 确定你知道主机代理商提供的服务器上的根目录哪个是你的。
- ❑ 写出从你的电脑传送文件到服务器的最佳途径和最佳工具。
- ❑ 现在，传送Starbuzz的“index.html”和“mission.html”文件到服务器的根目录。



返回休闲室主页……

这里是岔路的末尾了，我们将回到学习Web的高速公路上来。在服务器的根目录下应该有两个Starbuzz页面“index.html”和“mission.html”（如果没有，还是要好好看下面的内容）。

以上所述都做好后，用浏览器从互联网上接收并显示这些页面，难道这不让人开心吗?让我们告诉你如何在浏览器上输入正确的地址……



URL
关键的USA

你已经听过熟悉的“h” “t” “t” “p” “冒号” “斜线” “斜线”不计其数次了，但是它们是什么意思呢?首先，你在浏览器中输入的地址叫做URL（Uniform Resource Locators，统一资源定位符）。

用我们的话来说就叫网站地址，但是没有人会这么问，所以我们也用URL。下面是一个URL的实例：

```
http://www.starbuzzcoffee.com/index.html
```

URL的开头部分告诉你用来接收资源的协议。

第二部分是网站名。这时你应该知道了。

第三部分是从根目录到资源的绝对路径。

URL是一个全球性地址，用于定位网上的资源，包括HTML页、音频、视频，以及其他形式的网上内容。

另外，为了指定资源的位置，URL同时也包含用于接收资源的协议名。

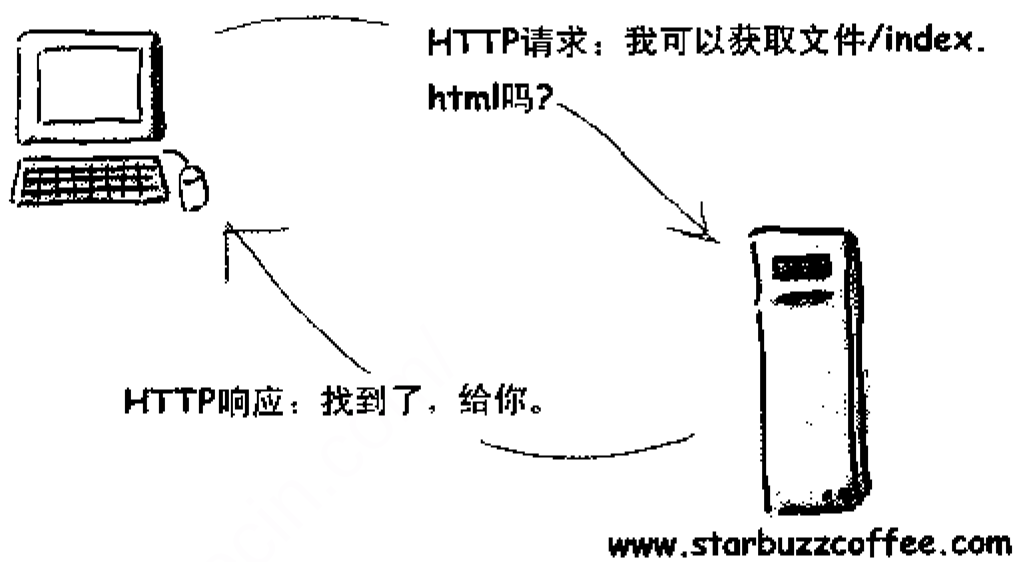
在网上寻找需要的东西，只需知道服务器主机名和到资源的绝对路径，就可以创建一个URL并用浏览器通过某些协议（通常是HTTP）接收到。



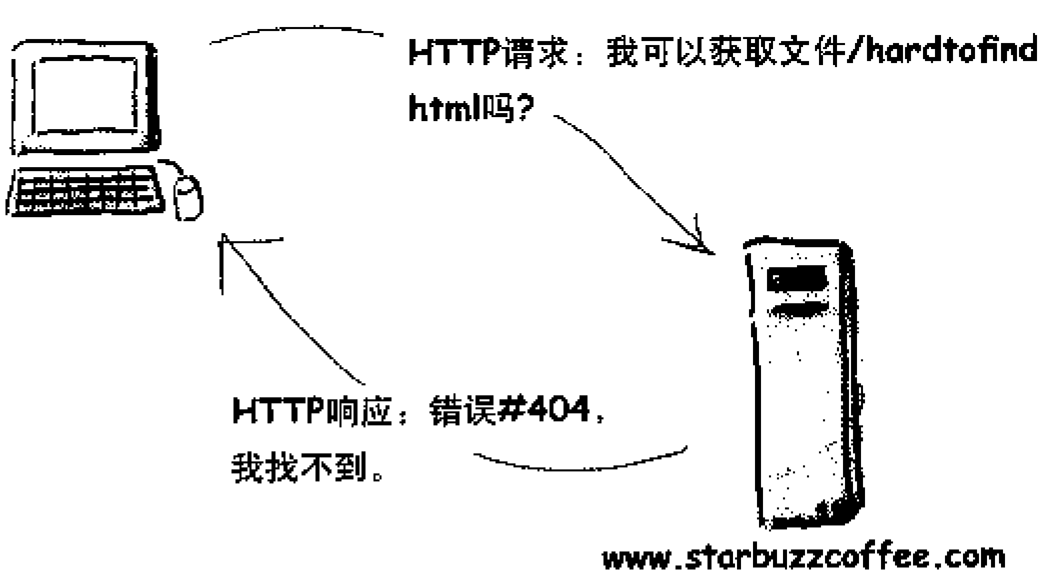
什么是HTTP协议?

HTTP是超文本传输协议（HyperText Transfer Protocol）。也就是说，是网上传送超文本文件的一致方法(协议)。“超文本文件（hypertext documents）”通常就是HTML页面，协议同时用于传送图片或者其他网页需要的文件。

HTTP是个简单的请求－响应协议。以下是它的工作原理：



所以每次你输入一个URL到浏览器的地址栏，浏览器都会用HTTP协议向服务器申请相应的资源。如果服务器找到了资源，则把它返回到浏览器，浏览器会显示它。如果服务器找不到会发生什么呢?





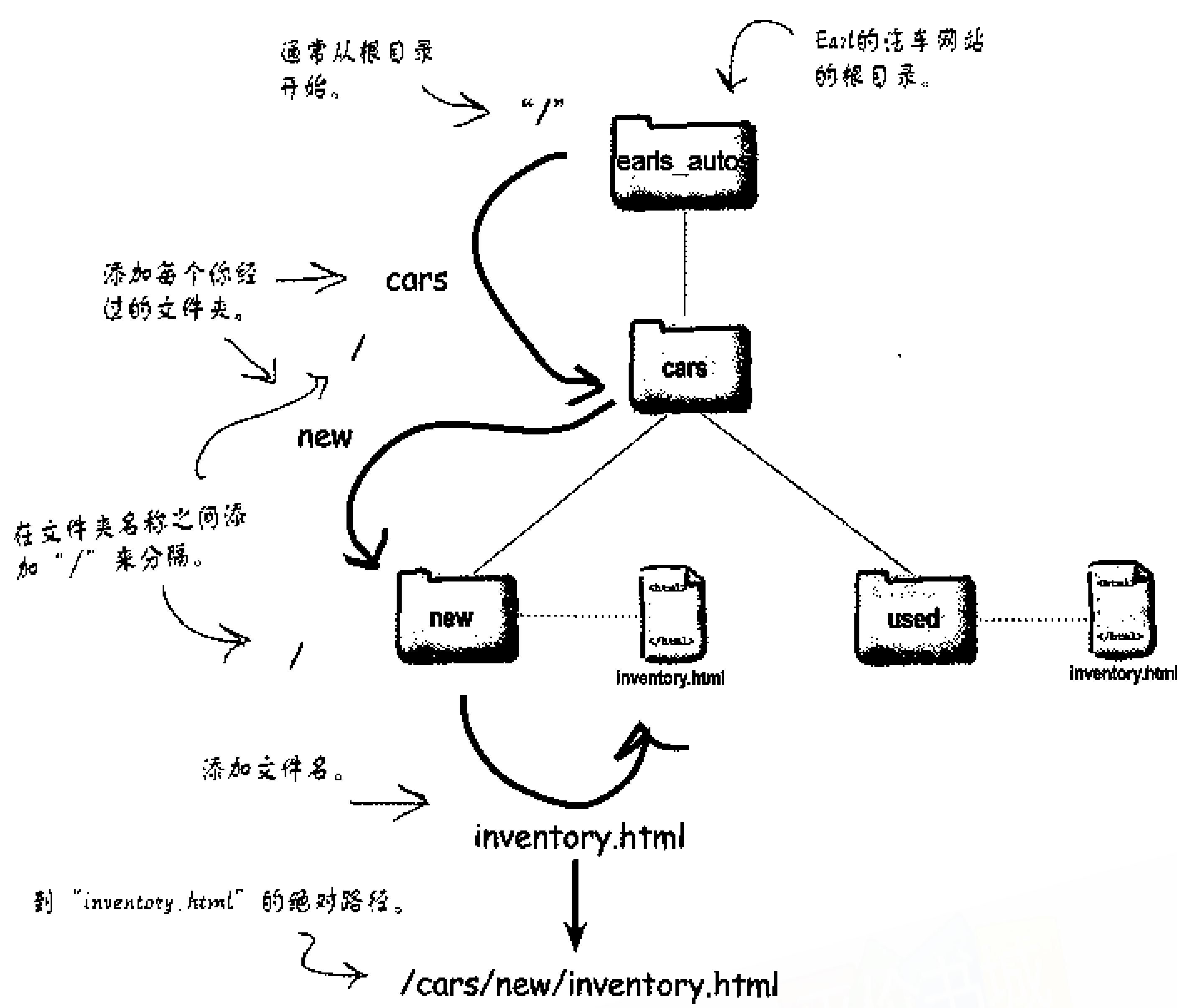
如果找不到资源，你会看到服务器返回给你的浏览器的熟悉的“404错误”。

什么是绝对路径？

上次我们讨论路径是在HTML中用<a>元素创建链接的时候。现在我们要考虑的路径是URL中的部分，叫绝对路径，就是在协议（http）和网站名（www.starbuzzcoffee.com）之后的部分。

绝对路径告诉浏览器如何从根目录去获得特定的页面或文件。拿Earl的汽车网站来说。如果你想看看在Earl的存货中有没有你定购的Mini Cooper，这时，你需要指出到“new”文件夹下的“inventory.html”文件的绝对路径。你要做的是追踪从根目录到“inventory.html”所在的“new”文件夹的路径。路径是由你所经过的所有文件夹组成的。

看起来就是根目录（用“/”表示），“cars”，“new”，最后是文件本身“inventory.html”。看看你是如何把它们放到一起的：



there are no Dumb Questions

问：绝对路径的重要性表现在哪里？

答： 绝对路径是服务器用于定位你要求的文件的。如果服务器没有绝对路径，它就不知道去哪里寻找。

问：我觉得各个部分都明白了（协议、服务器、网站和绝对路径），但是把它们联系在一起还是有些困难。

答： 把它们全部加到一起就是URL了，用URL你可以让浏览器从网上获取页面（或者其他类型的资源）。怎么获取？协议部分告诉浏览器获取资源的方式（大多数情况下是http）；网站名部分（由服务器名和域名组成）告诉浏览器从互联网上的哪台计算机获取资源；绝对路径告诉浏览器你在寻找什么页面。

问：我们学习了把相对路径放到<a>元素的href属性中。如果它们不是绝对路径，服务器是怎么找到它们的？

答： 问得好。当你点击一个相对链接时，浏览器会在后台根据相对路径和你所点击的网页的路径生成一个绝对路径。所以，所有Web服务器看到的都是绝对路径，这归功于你的浏览器。

问：如果我在HTML中输入绝对路径会对浏览器有帮助吗？

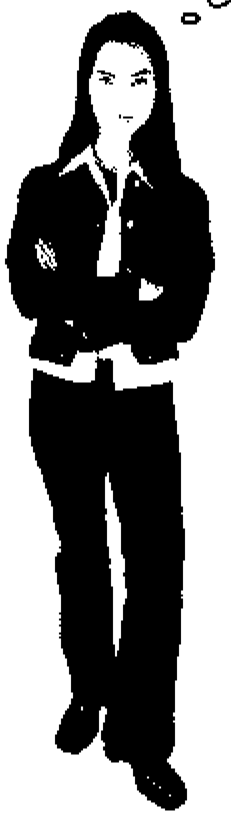
答： 又一个好问题！但是先别去考虑它，我们马上就会讨论。

Sharpen your pencil

让你久等了。是时候来测试一下新的URL了。先填好下面的空格然后输入URL（假设你还没有输入）。如果有什么困难，让你的主机代理商来解决吧。如果你没有主机代理商，填上www.starbuzzcoffee.com，然后输入到你的浏览器中。

__________ :// ____________________ ____________________

协议　　　　网站名　　　　绝对路径



记住，当我们谈到Web服务器或者FTP时，通常说的是目录而不是文件夹，其实两者都是一样的。

有的。我们还没谈到浏览器需要从服务器中获得一个目录而不是文件的情况。举例来说，浏览器可能这么请求：

http://www.starbuzzcoffee.com/images/

images目录在根目录下

或者

http://www.starbuzzcoffee.com/

根目录本身

当Web服务器接到这类请求时，它会尝试在目录中定位默认文件。通常默认文件名为“index.html”或者“default.htm”，如果服务器找到其中的一个文件，它会把此文件返回给浏览器显示。

所以，为了从你的根目录(或者其他目录)默认返回一个文件，给这个文件起名为“index.html”或者“default.htm”吧。

但是，你需要知道主机代理商希望你给默认文件起什么名字，因为这取决于他们的服务器的类型。



当然可以。当服务器接收到没有以“/”结尾的请求且有一个目录是这个名字时，服务器会为你添加一个斜线。所以如果服务器收到这样的请求：

http://www.starbuzzcoffee.com

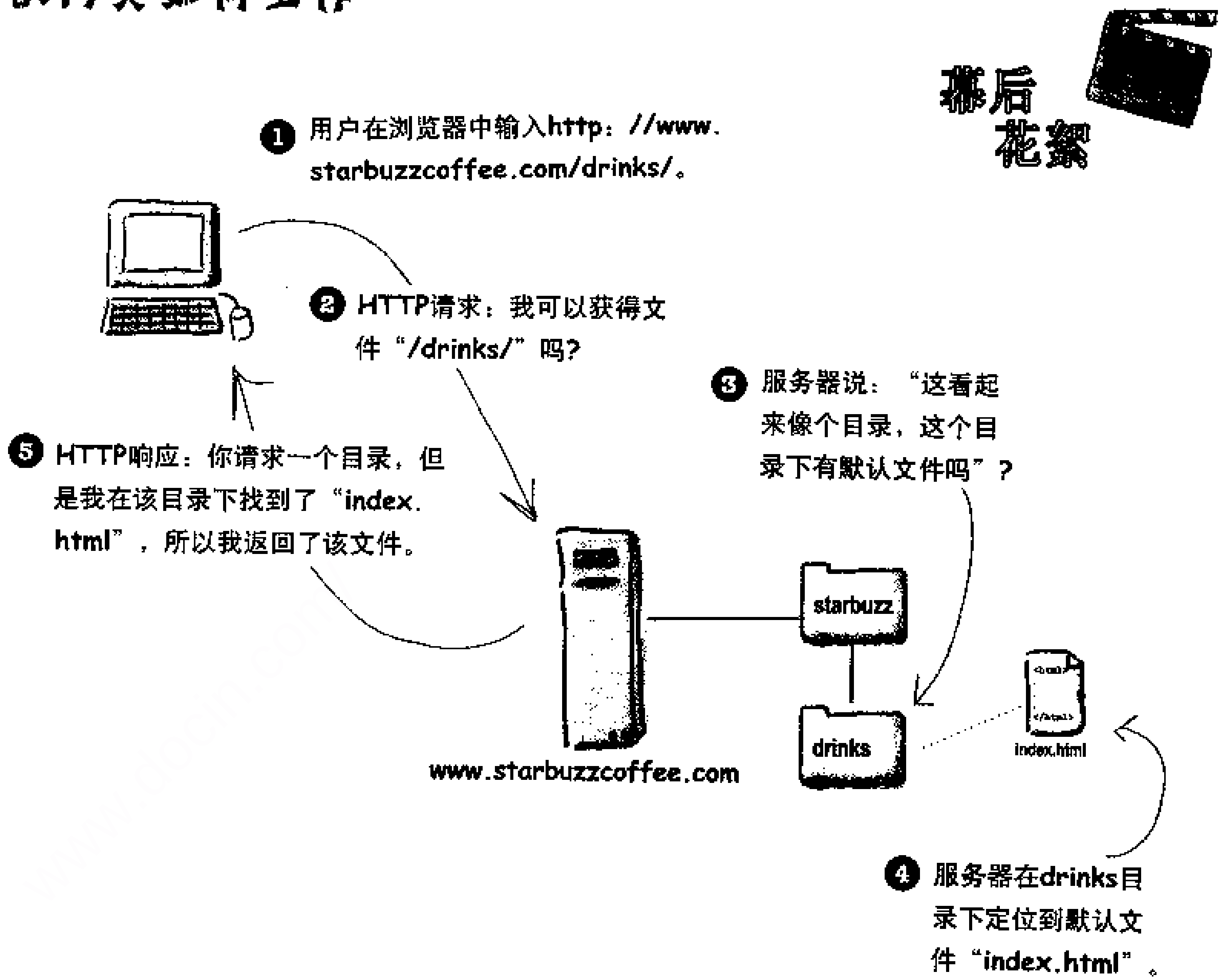
就会改成

http://www.starbuzzcoffee.com/

这会使得服务器去寻找默认文件，最后它会返回文件，好像你原来输入的是：

http://www.starbuzzcoffee.com/index.html

默认网页如何工作



there are no Dumb Questions

问：那是不是说每个来浏览我网站的人只要输入URL http://www.mysite.com就能看到我的“index.html”页面了?

答：对。不过，也有可能是“default.htm”，依你的主机代理商所用的服务器的类型而定。(注意“default.htm”中最后没有“l”。这是微软Web服务器的惯例。)

还有许多其他默认文件名，例如“index.php”，如果你开始用脚本编写网页的话就要用到它。这超出了本书的范围，但是，我们不提及并不代表你以后不会碰到它。

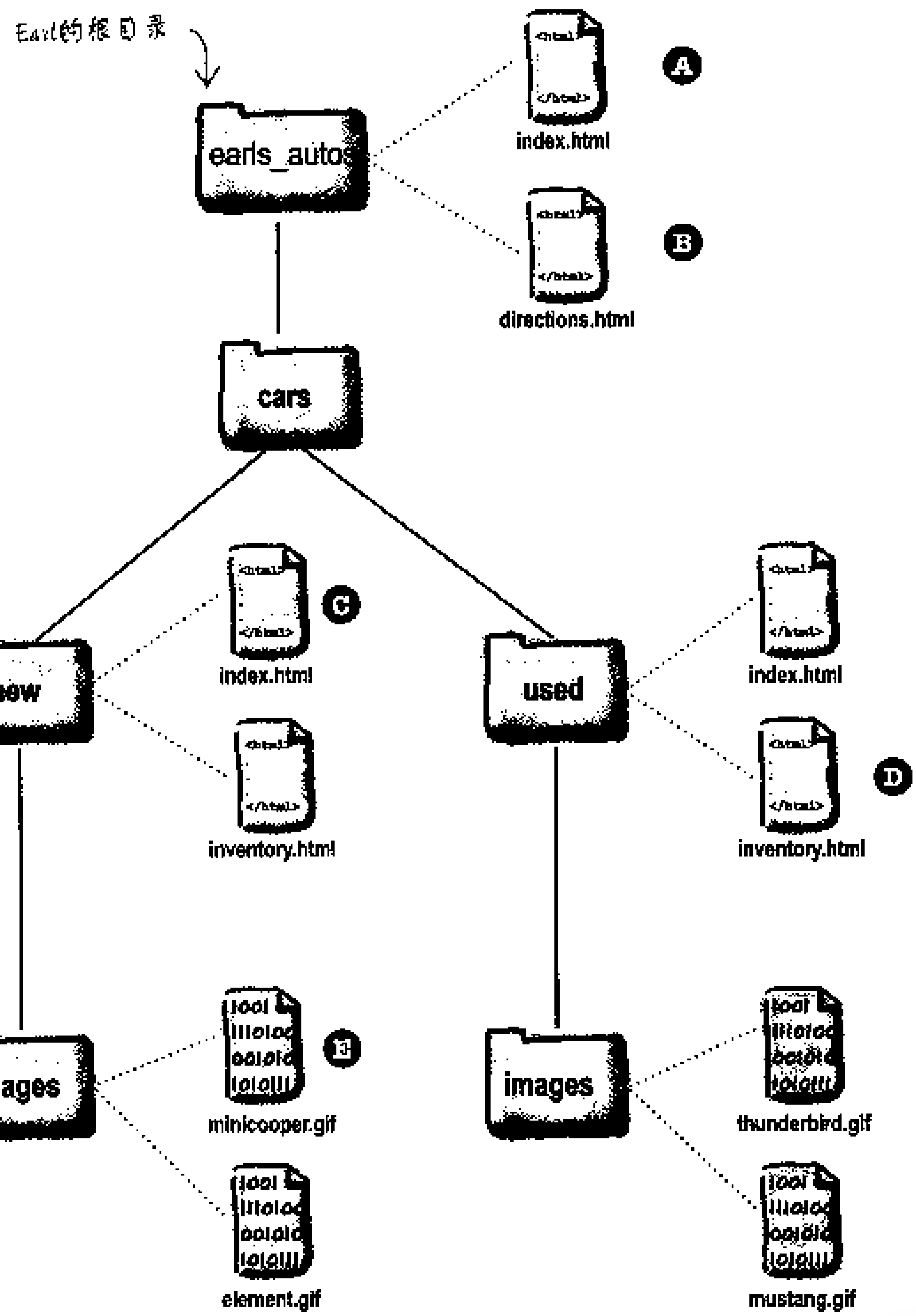
问：当我告诉别人某个URL时，是不是加上“index.html”比较好?

答：不是。通常不写它比较好。如果以后你换了个服务器，而那个服务器的默认文件名是“default.htm”怎么办?或者你开始用脚本编写网页并使用“index.php”呢?那你原来给出的URL就不合法了。

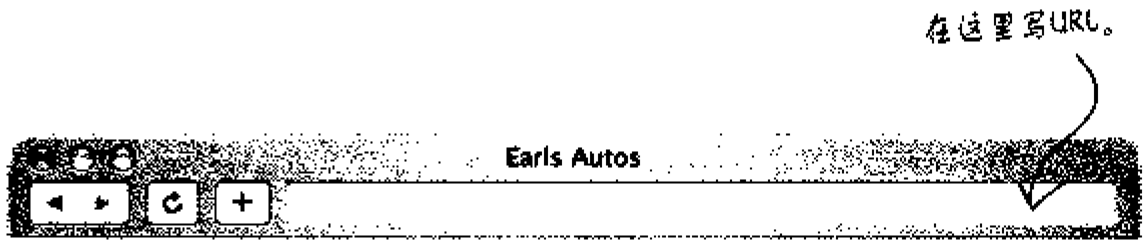


Earl的URL需要你的帮助

Earl可能知道Earl(伯爵)，但是他不知道U-R-L。他需要点帮助找出下列每个标着A，B，C，D，E的文件的URL。写出从www.earlsautos.com接收的相应文件的URL。



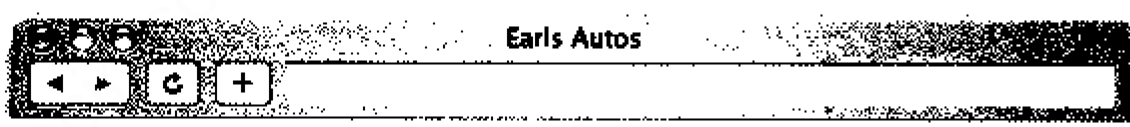
A


在这里写URL。
Earls Autos

B


Earls Autos

C

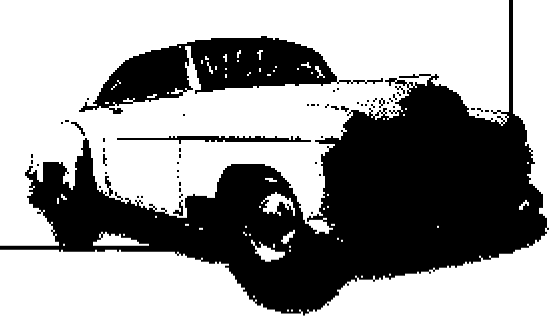

Earls Autos

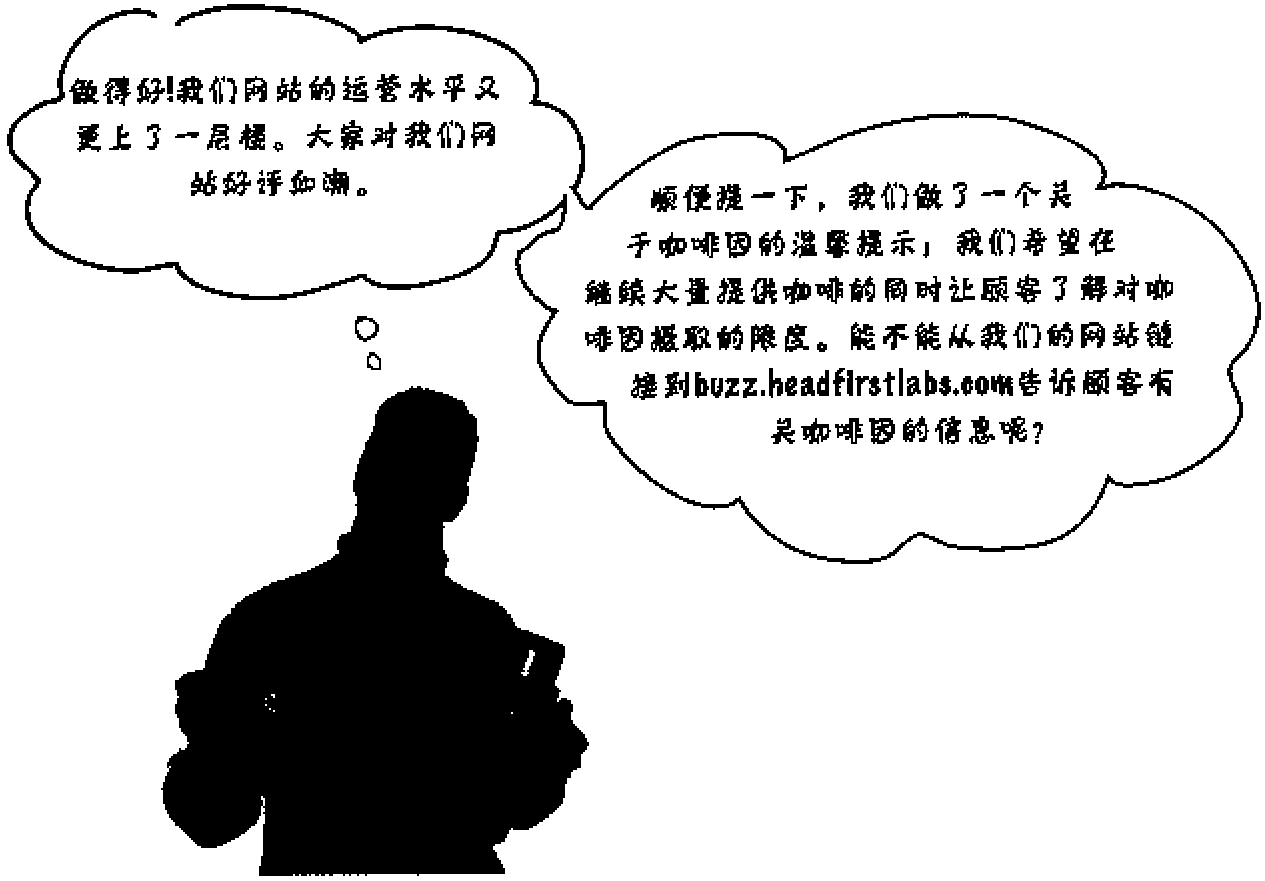
D


Earls Autos

E


Earls Autos





怎样链接到其他网页?

URL的功能并不局限于输入到浏览器，你还可以把它们应用到HTML中。同时，在提示中，Starbuzz的CEO有新的任务给你：从Starbuzz主页链接到http://buzz.headfirstlabs.com的咖啡因信息。可能你已经猜到，我们要在<a>元素中加入URL。下面是演示：

```
<a href="http://buzz.headfirstlabs.com">Caffeine Buzz</a>
```

很普通的<a>元素。

把URL放到href属性中。点击标签“Caffeine Buzz”，就会从buzz.headfirstlabs.com获得一个页面。

完成了！要链接到网上的资源，只需要该资源的URL，把它放在<a>元素中作为href属性的值。把它添加到Starbuzz“index.html”页面中并继续往下看。

链接到Caffeine Buzz

在“chapter4/starbuzz”文件夹中打开“index.html”，一直搜索到最底层目录。我们来添加两个新链接：一个是到“mission.html”的服务宗旨声明，另外一个是到Caffeine Buzz的链接。按照如下所示做修改，然后保存并用浏览器加载你的“index.html”。点击链接并体会Caffeine Buzz带给你的视觉享受。

```
<html>
    <head>
        <title>Starbuzz Coffee</title>
        <style type="text/css">
            body {
                background-color: #d2b48c;
                margin-left: 20%;
                margin-right: 20%;
                border: 1px dotted gray;
                padding: 10px 10px 10px 10px;
                font-family: sans-serif;
            }
        </style>
    </head>

    <body>
        <h1>Starbuzz Coffee Beverages</h1>
        <h2>House Blend, $1.49</h2>
        <p>A smooth, mild blend of coffees from Mexico,
           Bolivia and Guatemala.</p>

        <h2>Mocha Cafe Latte, $2.35</h2>
        <p>Espresso, steamed milk and chocolate syrup.</p>

        <h2>Cappuccino, $1.89</h2>
        <p>A mixture of espresso, steamed milk and foam.</p>

        <h2>Chai Tea, $1.85</h2>
        <p>A spicy drink made with black tea, spices,
           milk and honey.
        </p>
        <p>
          <a href="mission.html">Read about our Mission</a>
          <br>
          Read the <a href="http://buzz.headfirstlabs.com">Caffeine Buzz</a>
        </p>
    </body>
</html>
```

这是到“mission.html”文件的链接。这里用相对路径链接到该文件。

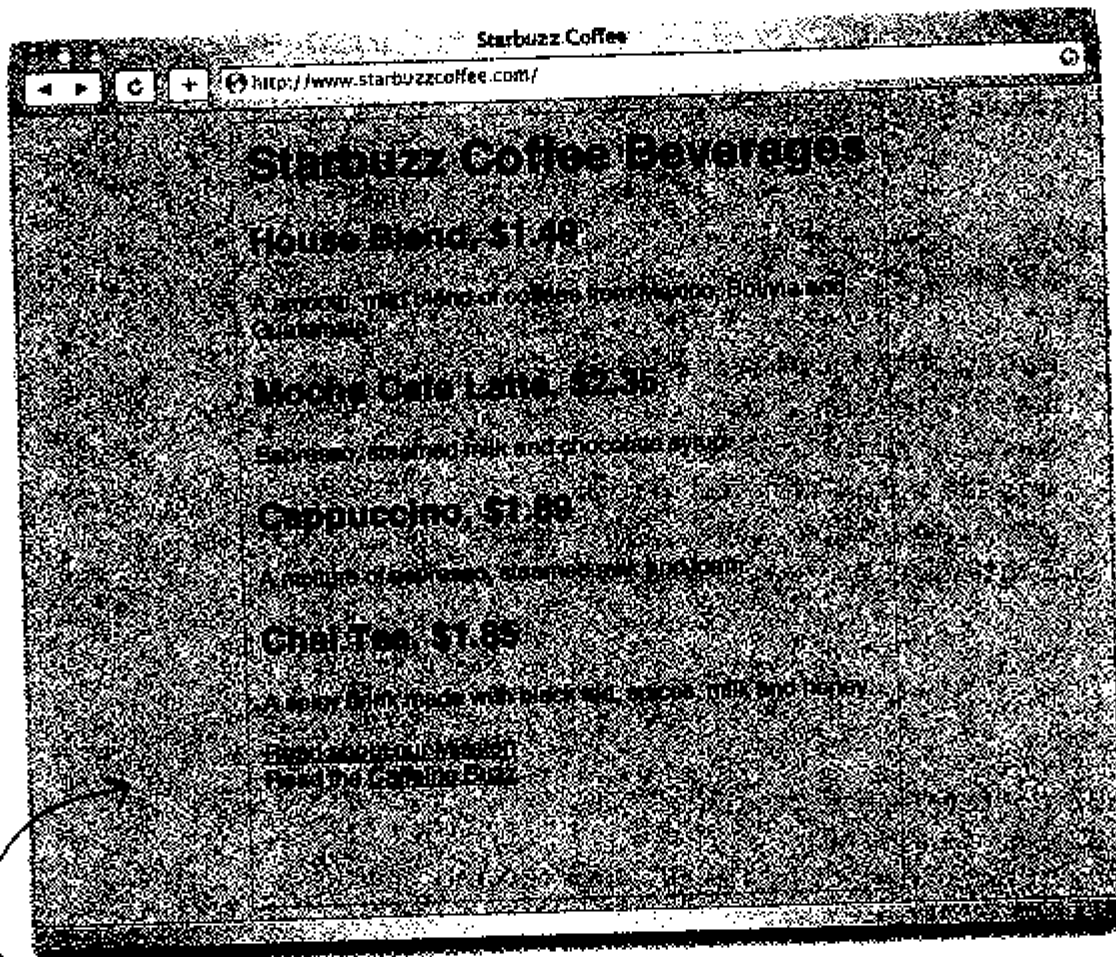
添加一个
把两个链接分成两行。

这是添加的到buzz.headfirstlabs.com页面的链接。

通过将链接和文本组织在一个段落中来添加结构。

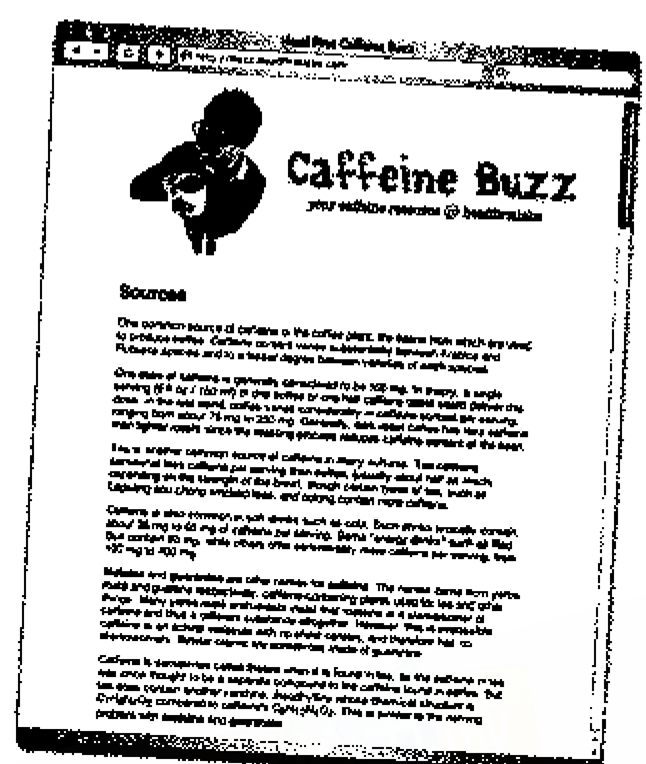
现在测试一下……

这是个新网页，里边含有我们设计好的新链接。



这里是新链接。注意，我们只链接了词组"Caffeine Buzz"，所以它看起来和其他链接不同。

当你点击链接时，浏览器会发送一个http请求到buzz.headfirstlabs.com并显示结果。



there are no Dumb Questions



问：有两种途径可以链接到页面：相对路径和URL。是吗？

答： 相对路径只能用于链接同一网站内的页面，而URL通常用来链接到其他网站。

问：那我全用URL来链接站内站外的页面不是更简单吗?为什么不这样做?

答： 当然，能这样做，但是诸多因素会使你放弃这种做法。一个问题是，当你的网页中有很多URL的时候你很难控制好：它们都很长，难编辑，还会使HTML的可读性变差(特别是对你这个网页编写者而言)。

同时，如果你的网站只有本地的URL，而当你需要移动网站或者给网站改名的时候，你就得更改所有的URL来对应新的位置。如果使用相对路径，只要你的页面还在原来的文件夹里，就不需要对<a>元素href属性做任何改变，因为链接都是相对的。

所以，使用相对路径来链接你的本地网页，使用URL来链接其他网站的页面。

问：没有其他的协议吗?我开始使用服务器之前总是看见"file：//"。

答： 对，好眼力。file协议是浏览器从你的电脑中读取文件时用的。举例来说"file：///chapter4/starbuzz/index.html"，告诉浏览器文件"index.html"在路径"/chapter4/starbuzz/"下。这个路径会因操作系统而异。

有件很重要的事要注意，在文件中输入URL要三个斜线，而不是像HTTP那样的两个斜线。这样来记：如果你删去一个HTTP URL的网站名，同样也会有三条斜线留下。

问：还有其他协议吗?

答： 有，许多浏览器可以支持FTP协议来接收文件，还有mail协议，通过E-mail来传送数据。HTTP是最常用的一种。

问：我见过这样的URL：http://www.mydomain.com：8000/index.html。为什么有个"：8000"呢?

答： "：8000"是你在HTTP URL中输入的可选端口。对于端口，你可以这样想：如果网站名像个地址，那么端口就像该地址的邮箱号码(比如在复合式公寓)。通常网上的所有东西都传送到一个默认端口（通常是80），但有时Web服务器会设定一个不同的端口（像8000）来接收请求。你可能会在测试服务器中看到它。一般的Web服务器在大多数情况下用端口80来接收请求。如果不专门指定，则默认为80。

相对与绝对之争

PlanetRobots有限公司现在面对着一个难题：为它下属的两个部门——PlanetRobot Home和PlanetRobot Garden各建一个网站，于是决定跟两家公司签约来完成这项工作。RadWebDesign，一个看起来经验老到的公司，接受了Home部门的独立网站设计并只用URL来做内部链接（毕竟它们更复杂，当然更加优秀）。CorrectWebDesign，一个经验不够丰富但受过良好教育的公司，接受了PlanetRobot Garden网站的设计任务，并在网站内部的页面之间使用相对路径来链接。

五分钟之谜



当两个公司的工作都接近尾声的时候，PlanetRobots给他们一个紧急消息："我们被告商标侵权，所以现在要改域名为RobotsRUs。我们的新Web服务器将改成www.robotsrus.com。" CorrectWebDesign只用5分钟做了些小修改，然后就整理好等着RobotsRUs公司总部的新网站发布会。而另一边，RadWebDesign一直修改网页到凌晨4点，幸亏还是赶上了发布。然而天有不测风云，在发布会的演示中，意外发生了：当RadWebDesign的工作组组长展示网站并点击一个链接的时候，居然出现了错误"404-Page Not Found(找不到页面)"。RobotsRUs的CEO不屑地建议RadWebDesign应该改名为BadWebDesign，并向CorrectWebDesign咨询能否修复Home网站上的错误。

发生了什么事？为什么只是由于Web服务器改了名，RadWebDesign就会弄得这么糟？

网页完善

你明白什么是“Web生涯”了吗？你已经做好了所有Starbuzz的CEO所要求的东西并有了一个外形鲜明的网站。

但是你还不能停下来。你需要不断地完善你的网站，使它由良好上升到优秀。在本书的剩余章节，你会学到许多使网站“闪闪发亮”的方法，让我们先从改进链接开始吧。

添加标题使得链接更易理解

如果有一种方法，可以在你点击链接时提供更多信息，那多好！这对视力有障碍、使用屏幕读取器的用户来说尤为重要，因为它们经常不希望把整个URL读出来（“h” “t” “t” “p” “冒号” “斜线” “ 斜线” “w” “w” “w” “点”），链接标签通常只是给出最简短的描述，例如“Caffeine Buzz”。

<a>元素中有个叫做**title**的属性，正是为这个目的而生的。有些人会因为有个在**<head>**中使用的叫做**<title>**的元素而跟这个属性名混淆。它们有共同的名字，因为它们是相关的——通常我们建议**title**的值应该和你链接的网页上的**<title>**元素值一致。这不是硬性要求，但这样对你更方便，在**title**属性中有更多的相关描述。

这里教你如何把**title**属性添加到**<a>**元素中：

```
Read the <a href="http://buzz.headfirstlabs.com"
          title="Read all about caffeine on the Buzz">Caffeine Buzz</a>
```

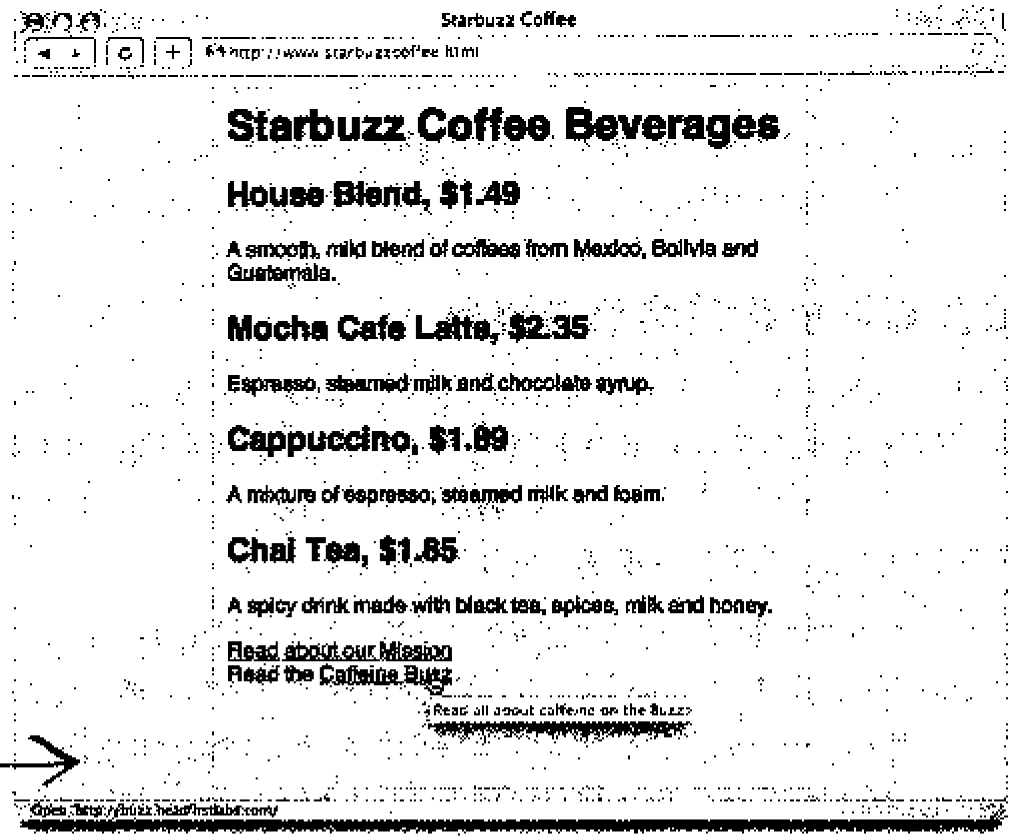
title元素的这个值是对你链接的网页的文字描述。



既然有了title属性，我们就来看看你的浏览者是如何使用它的。不同浏览器对title有不同的使用方法，但是大多只是显示一个工具条。如上更改“index.html”文件并重载网页，看看它在浏览器中如何运行。

测试标题……

在大多数浏览器中，当你把鼠标悬停在链接上时，标题是用工具条提示的形式显示的。为视觉有障碍的用户设计的浏览器会大声念出链接标题。



标题在大多数浏览器中显示为“工具条”。当你把鼠标悬停在链接上并停留一段时间的时候，就会看到工具条。

完善链接HeadFirst指南



这是些完善链接所要记住的技巧。

a 让链接标签简短些。不要用整个句子或者大段文字来做链接标签。通常来说，几个词就好了。在title属性内添加附加信息。

a 注意链接标签要有提示性。绝不要使用像“click here(点击这里)” “this page（这一页）”这样无意义的标签。用户会先尝试从网页中寻找链接，然后再浏览网页。所以，提供有意义的链接会增进网页的可用性。阅读链接，并以此来测试你的网页：它们有意义吗?或者你需要通读它周围的文字吗?

a 避免把链接紧挨在一起：链接靠得太近的话用户不易区别。

练习

打开你的Starbuzz“index.html”文件并给到“mission.html”的链接添加标题，内容为“Read more about Starbuzz Coffee's important mission”。注意我们还没有简写这个标签。把链接标签简写为“our Mission”。对照本章末的答案，测试你的修改。

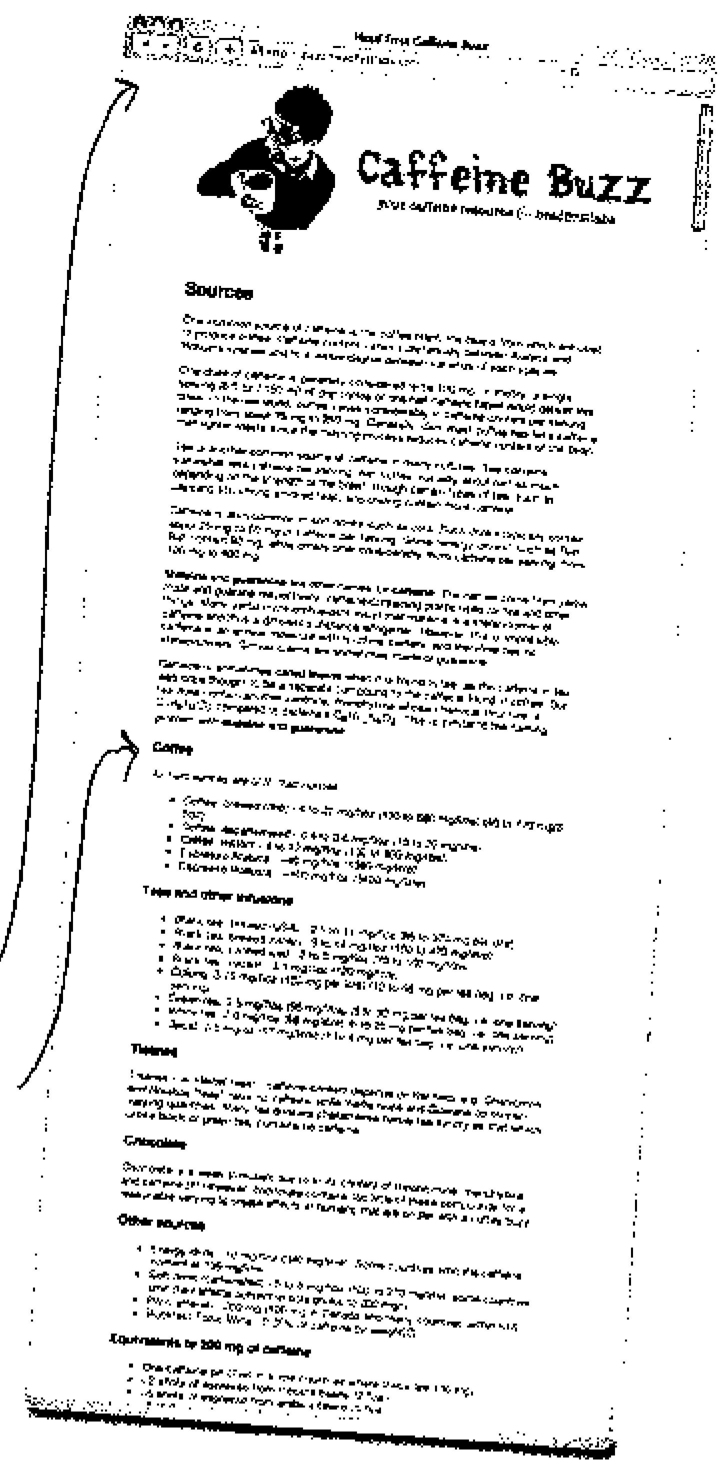


链接到一个页面

当你链接到一个其他的网页，浏览器总是从顶部开始显示网页。

但是，CEO要求你链接到页面的一个特定区域——coffee专区。

听起来不可能?别担心，这里是Head First——我们有许多神奇的小技巧。怎么做呢?嗯，其实我们还没有告诉你有关<a>元素的全部内容。告诉你，<a>元素有两种作用：你已经见识过它的跳跃性，从一个页面到另外一个页面；但它还可以在页面中设置登录点，或者链接的目的地。



使用<a>元素创建目的地

当你使用**<a>**元素创建目的地时，我们称它为“目标锚”。创建一个目标锚很简单，只需以下三步：

1. 找到网页中需要创建登录点的地方。它可以是文本中的任意位置，不过通常是文章标题的一小部分。
2. 用**<a>**元素把文本包围起来。
3. 选择目的地的标识名，比如“coffee”或“summary”或者“bio”，并在**<a>**元素中插入**id**属性。

我们来做个测试吧。假如你希望能链接到Srarbuzz页面上的Chai Tea（香辣奶茶）这一项。应该这样做：

这里是“index.html”中关于Chai标题和描述的片断。

```
<h2>Chai Tea, $1.85</h2>
<p>A spicy drink made with black tea, spices, milk and
honey.</p>
```

根据以上三步，我们会得到：

在文本之前添加<a>开始标记。

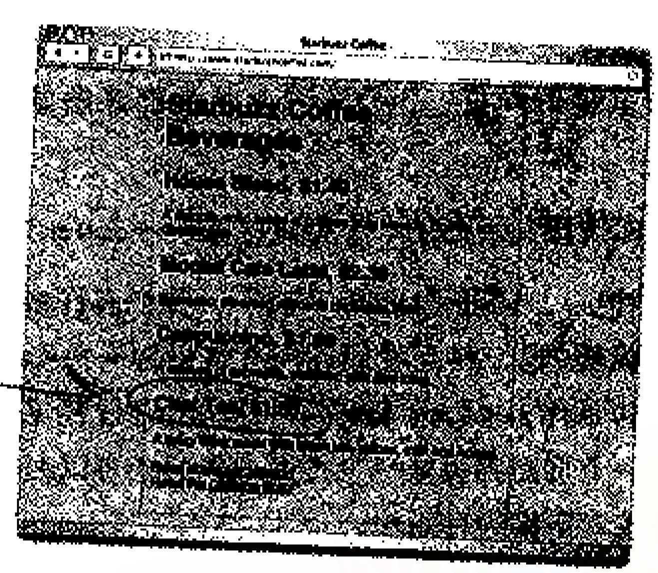
给出目的标识符“chai”。

用结束标记结束元素。

确保<a>元素完全嵌套在<h2>元素中。

```
<h2><a id="chai">Chai Tea, $1.85</a></h2>
<p>A spicy drink made with black tea, spices, milk and
honey.</p>
```

你已经创建了一个目标锚指向“index.html”页中的Chai Tea标题。



如何链接到目标锚

你已经学会了如何用相对链接或者URL来链接到网页了。无论是哪种方式，你都要在链接后边添加#，后边再加目标锚标识符。这样能更明确地链接页面中的目标锚。所以如果你想从Starbuzz Coffee 网页链接到“chai”目标锚，你需要这样改写<a>元素：

```
<a href="index.html#chai">See Chai Tea</a>
```

然而，用目标锚链接到Chai Tea不是个让人印象深刻的例子，因为整个网页太小，恰好浏览器一页就能显示了。取而代之，让我们链接到http://buzz.headfirstlabs.com的Coffee区域。

目标锚的主要好处是链接到比较长的页中的特定位置，以使浏览者不用滚动页面来寻找。

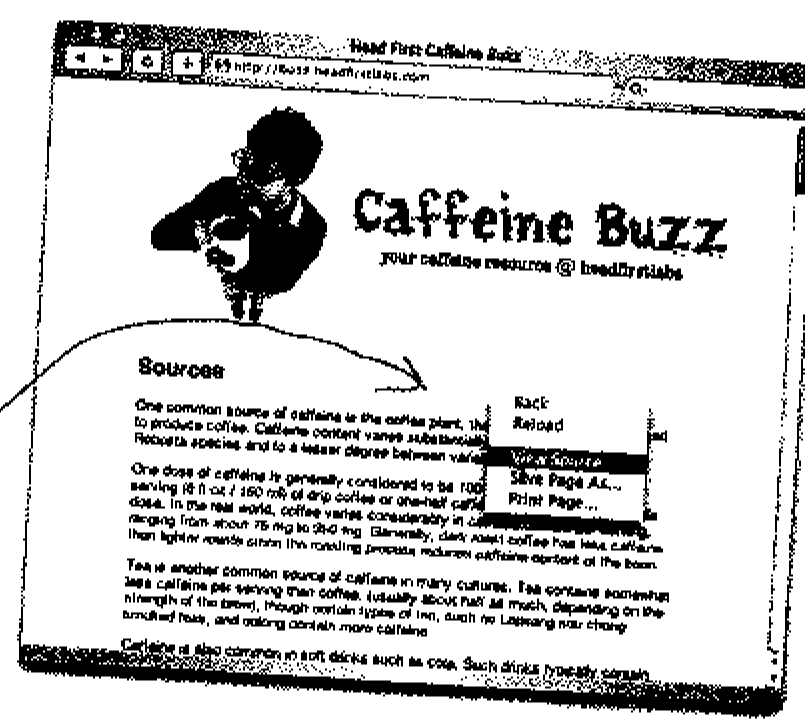
这里是你要做的：

1. 指出目标锚的id。
2. 改变Starbuzz Coffee中“index.html”的原有<a>元素，使其指向目标锚。
3. 重载“index.html”页并测试链接。

找到目标锚

为了找到目标锚，你需要到buzz.headfirstlabs.com页看看并浏览他们的HTML。怎么看?大多数浏览器都有“View Source（查看源文件）”选项。访问网页，等装载完成后，选择“View Source”选项，你会看到页面的标记。

在大多数浏览器里，你可以通过单击右键找到“View Source”，同样可以在浏览器菜单中找到“View Source”，通常是在“View（查看）”菜单下。



现在你已经弄到他们的HTML了……

下翻直至你看到Coffee专区，看起来就像这样：

Coffee Buzz页的片断。

```
This is similar to the naming problem
with <b>mateine</b> and <b>guaranine</b>.
</p>

<h3><a id="Coffee">Coffee</a></h3>
<p>
<i>All fluid ounces are U.S. fluid ounces.</i>
</p>
```

这是Coffee专区。在下面一段的开头你可以看到标题。

啊哈，这是目标锚，名为“Coffee”。

对“index.html”链接的重新修整

现在要做的是重新访问到Caffeine Buzz的链接并添加目标锚名，如下所示：

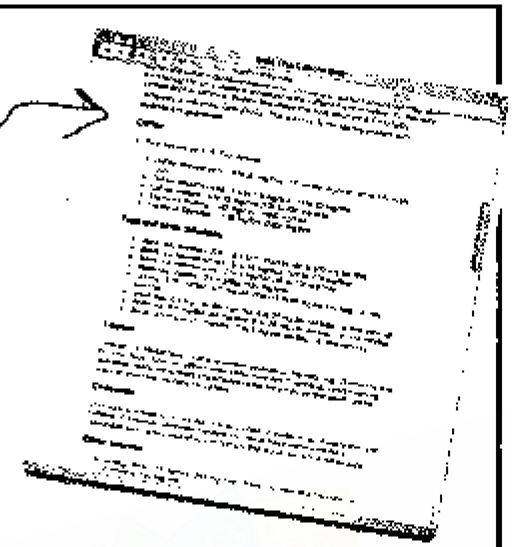
这是Starbuzz“index.html”的片断。

在href中添加#和随后的目标锚id。

```
Read the <a href="http://buzz.headfirstlabs.com#Coffee"
            title="Read all about caffeine on the Buzz">Caffeine Buzz</a>
```

练习

对你的Starbuzz“index.html”文件做同样的修改。重载并点击“Caffeine Buzz”链接。你就可以直接转到Caffeine Buzz首页的Coffee专区了。



there are no Dumb Questions

问：当我给一个元素添加两个属性时，它们之间的顺序重要吗?举例来说，title属性应该跟在href后边吗?

答： 在任何元素中，属性顺序都不重要(如果重要的话我们就得头疼了)。所以，你可以用任意你喜欢的顺序。

问：通常我使用<a>元素的时候浏览器会在文本下加下划线，但是我使用id属性代替href之后就没有了。

答： 对。当你使用id属性时不会对<a>元素所包围的文本的外观有任何影响。记住，目标锚（<a>和id属性）的目的只是标识页中的位置，而不是创建链接。所以没有必要特殊显示。

问：为什么称它为锚?它看起来像锚吗?

答： 我们不得不承认：锚这个名字起得不好，在你之前曾有不计其数的人觉得困惑。我们都没法给你一个确切的说法来让你明白它怎么是个锚。我们都习惯用这个名字了，现在你已经知道它是干什么的了，而在这之前你甚至不能给这个名字一个另外的解释。

问：就算名字起得差，为什么要用同一个元素做不同的事?为什么不分开链接和定位的元素?

答： 可以这样想：你拥有的是从某处到某处的链接。带href属性的<a>元素提供了一条途径来描述“从某处”，而“到某处”通常意味着到其他网页的开头处——换句话说，你到达的是某处的任意位置。

问：我发现在锚的id名中，你使用的是小写的“chai”而Caffeine Buzz的“Coffee”中用的“C”是大写的。这会有影响吗?

答： 你可以在id属性中混用大写和小写字母。但是在href属性和目标锚id中，要确保字母大小写一致(这就是为什么通常容易将这些名字全部小写的原因)。如果不一致，链接就不能在每个浏览器中正常工作。

问：我可以在某文件中添加一个到该文件某目标锚的链接吗?

答： 当然可以了。其实，在网页的顶部指定一个目标锚“top”，在网页的底部有一个名叫“Back to top”的链接是很正常的。我们经常在长文件中制作内容表。举例来说，在相同的页面中链接到顶部的目标锚，你可以写成这样：Back to top。

问：如果一个网页不能提供目标锚而我需要链接到该页的特定位置，我应该怎么办?

答： 那就没有办法了。如果没有目标锚你就不能让浏览器定位到网页的特定位置。你可以尝试去联系网页制作者并要求他们添加一个(如果他们不会，你来告诉他们怎么做)。

问：我能不能有个像“Jedi Mindtrick”这样的id呢？id只能有一个词吗?

答： 大多数浏览器一般会要求你的id以一个字母（A-Z或者a-z）开始，后边跟着字母、数字、连字符、下划线、冒号或者句点。因为不能使用空格，所以不能有“Jedi Mindtrick”这样的名字；但是这不是很严格的限制，因为你可以起“Jedi-Mindtrick”，“Jedi_Mindtrick”，“JediMindtrick”之类的名字。

问：我怎么告诉其他人他们可以链接什么目标锚呢?

答： 没有确定的方法来这么做，实际上“View Source”选项是最古老实用的方法了。

相对与绝对之争

解开五分钟之谜

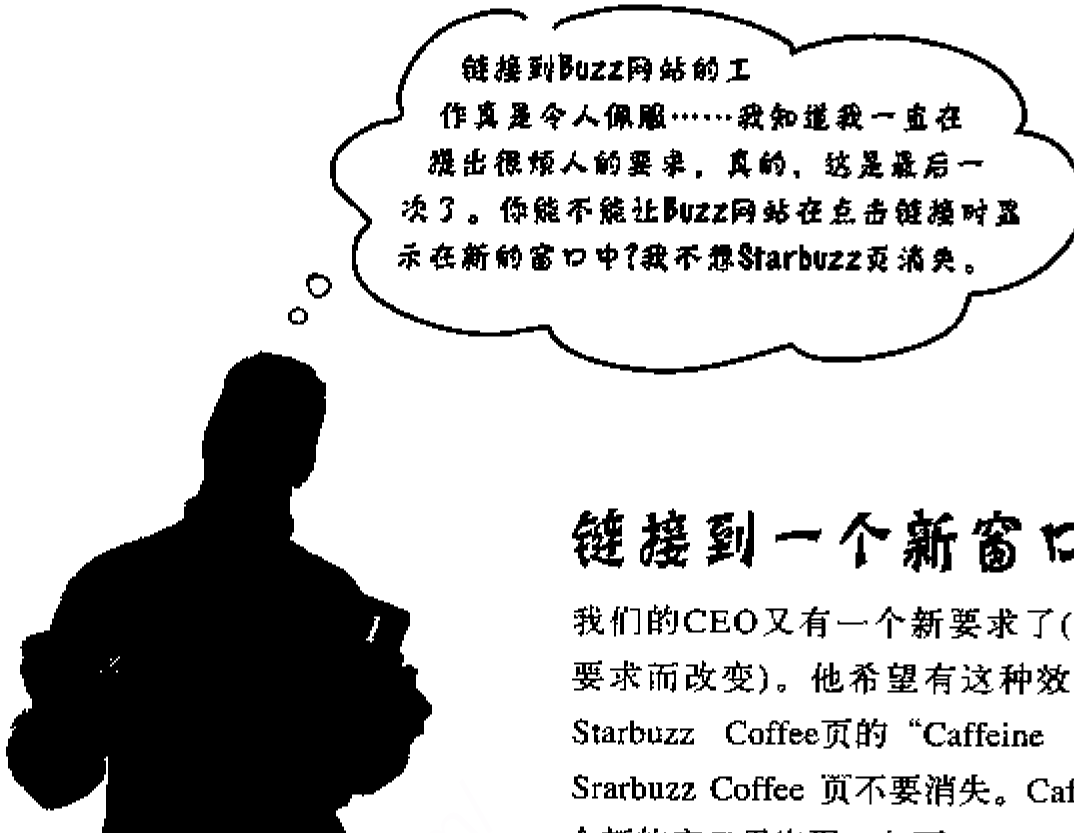


RadWebDesign为什么会把演示弄砸呢?因为他们为其href使用了URL而不是相对链接，他们必须把每一个到http://www.planetrobots.com的链接改成http://www.robotsrus.com。谁能保证自己永不出错?在凌晨3点，有人开了个小差，不小心输入成了http://www.robustsru.com(命中注定，CEO在演示时恰巧点击了这个链接)。

喔喔喔，有人在名字后边忘记了一个“S”

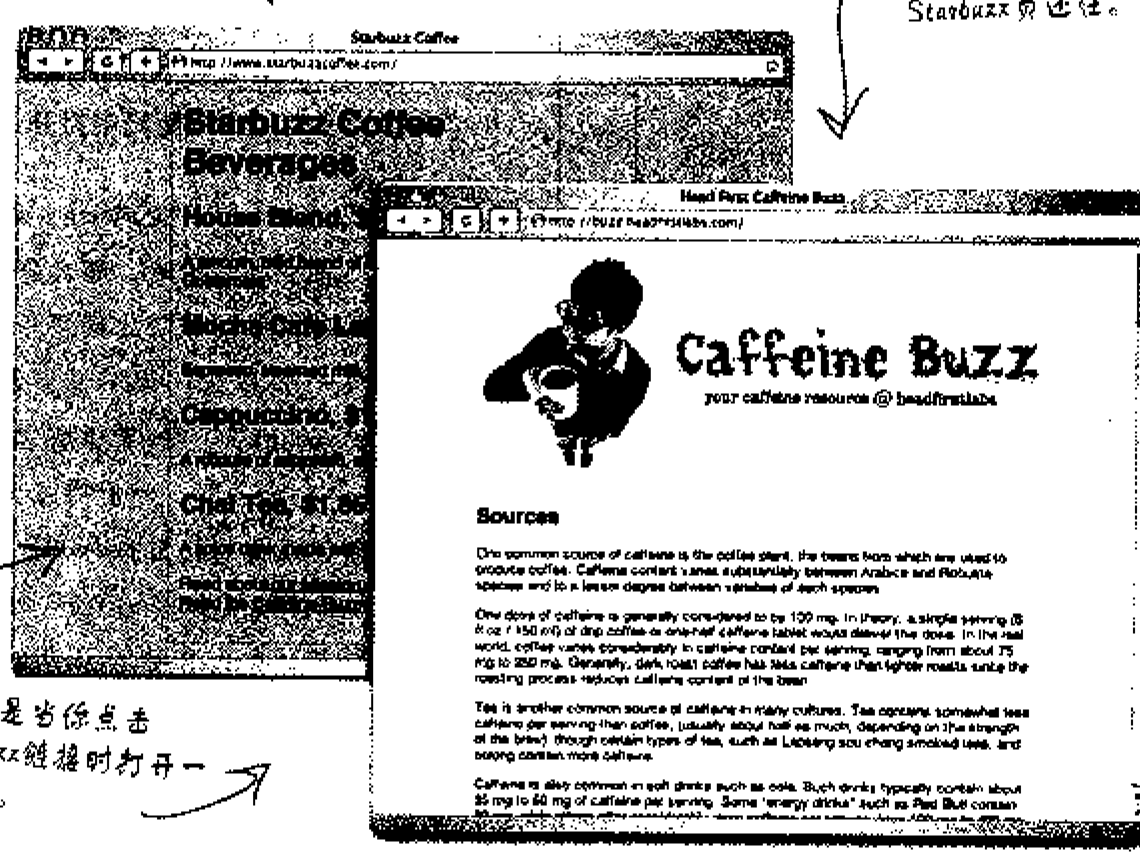
另一方面，CorrectWebDesign在内部链接使用了相对路径。举例来说，从公司的服务宗旨页面到产品页的链接，<a href=“../procducts.html”>，无论网站是叫PlanetRobots还是RobotsRUs，都能正常工作。所以，CorrectWebDesign要做的只是在某些页面更改公司名就好了。

在满场嘘声中RadWebDesign沮丧地离开了演示会，而CorrectWebDesign在此之后获得了更多业务。但是，故事还没有结束。在演示会后RadWebDesign访问了一些咖啡馆和书店，挑了几本HTML和CSS方面的书攻读，决心要挽回面子。后来怎样了？请留意后边章节中的“蛮力对样式的案件”。



链接到一个新窗口

我们的CEO又有一个新要求了(网站经常会因要求而改变)。他希望有这种效果：当你点击Starbuzz Coffee页的“Caffeine Buzz”链接时，Srarbuzz Coffee 页不要消失。Caffeine Buzz在一个新的窗口里出现，如下：



使用target(对象)打开一个新窗口

如果要在一个新窗口中打开一个网页，你需要告诉浏览器那个新窗口的窗口名。如果没有指定具体的窗口，浏览器将在原来的窗口中显示网页。可以通过在**<a>**元素中添加**target**属性来告诉浏览器使用不同的窗口。**target**属性的值就是浏览器页的“对象窗口”。如果把对象值指定为“_blank”，浏览器将总是打开新窗口来显示网页。让我们仔细看一下：

```
<a target="_blank" href="http://buzz.headfirstlabs.com"
   title="Read all about caffeine on the Buzz">Caffeine Buzz</a>
```

target属性告诉浏览器在哪里打开href属性指定的网页。如果没有target，浏览器会在同一个窗口打开链接。如果target(属性)是“_blank”，浏览器会在新的窗口打开链接。

练习

打开Starbuzz中的“index.html”文件。给链接到Caffeine Buzz页面的<a>标签添加**target**属性。试着修改一下，是不是在新窗口中显示了？

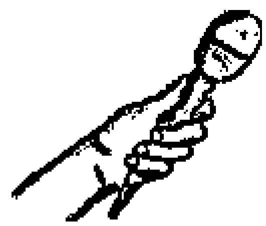
BRAIN POWER

想想使用target属性在新窗口中打开网页的优点和缺点。

there are no Dumb Questions

问： 如果有多个<a>元素含有对象(target)呢？如果已经打开一个“_blank”新窗口了，是在这个窗口中打开，还是在另外一个新的“_blank”窗口中打开？

答： 如果全部<a>元素中的对象名都是“_blank”的话，那么每个链接都会在新的空白窗口中打开。你问得很好，因为它告诉了我们一个要点：你不需要把所有对象都起名为“_blank”。如果你给它起另外一个名字，例如“coffee”，所有对象名为“coffee”的链接都会在相同的窗口打开。原因是当你给对象一个特定名称如“coffee”时，就为显示链接页的新窗口起好名字了。“_blank”是个特例，它告诉浏览器总是使用一个新窗口。



target属性揭密

本周访问：
使用target被认为很糟糕？

Head First：你好，target，非常高兴你能接受我们的采访。

Target属性：我很高兴来到这里。很高兴你们对我还感兴趣。

Head First：为什么这么说呢？

Target属性：呃，老实说，我没以前那么流行了。

Head First：你为什么这么想呢？

Target属性：我想是因为用户想控制打开的窗口。他们通常不喜欢不时地弹出新窗口。

Head First：呃，那的确很烦人。我们听到过这样的抱怨：由于屏幕上显示过多的窗口，他们找不到最初的页面了。

Target属性：处理窗口看起来并不难……只要点击那个小小的关闭按钮。这也很难吗!?

Head First：对，但是如果用户不知道新窗口弹出了就会产生混淆。有时新窗口完全覆盖了旧窗口，很难说发生了什么。这样谁都会混淆，尤其是视觉有障碍的人。

Target属性：哦，我还没这么想过呢。

Head First：嗯，想想吧：如果有人放大了浏览器窗口，而且整个新窗口覆盖在他们正在读的网页上面，对他们造成的困扰一定不小。如果不留意看屏幕，甚至都不知道发生了什么事。

Target属性：我想是的。这对使用屏幕读取器的人来说可能也是困难的。

Head First：对。有些屏幕读取器会在打开新窗口时播放声音，但有的会完全忽略新窗口，有的会直接跳到新窗口。无论哪种方式，总会有人因为看不到发生了什么事而困惑。而且，因为网页是在一个新窗口中，所以没法使用返回键回到原始窗口。

Target属性：[苦笑]我开始明白我为什么不如当年流行了。

Head First：不要这么沮丧，有时能用新窗口打开也是好事，不是吗？

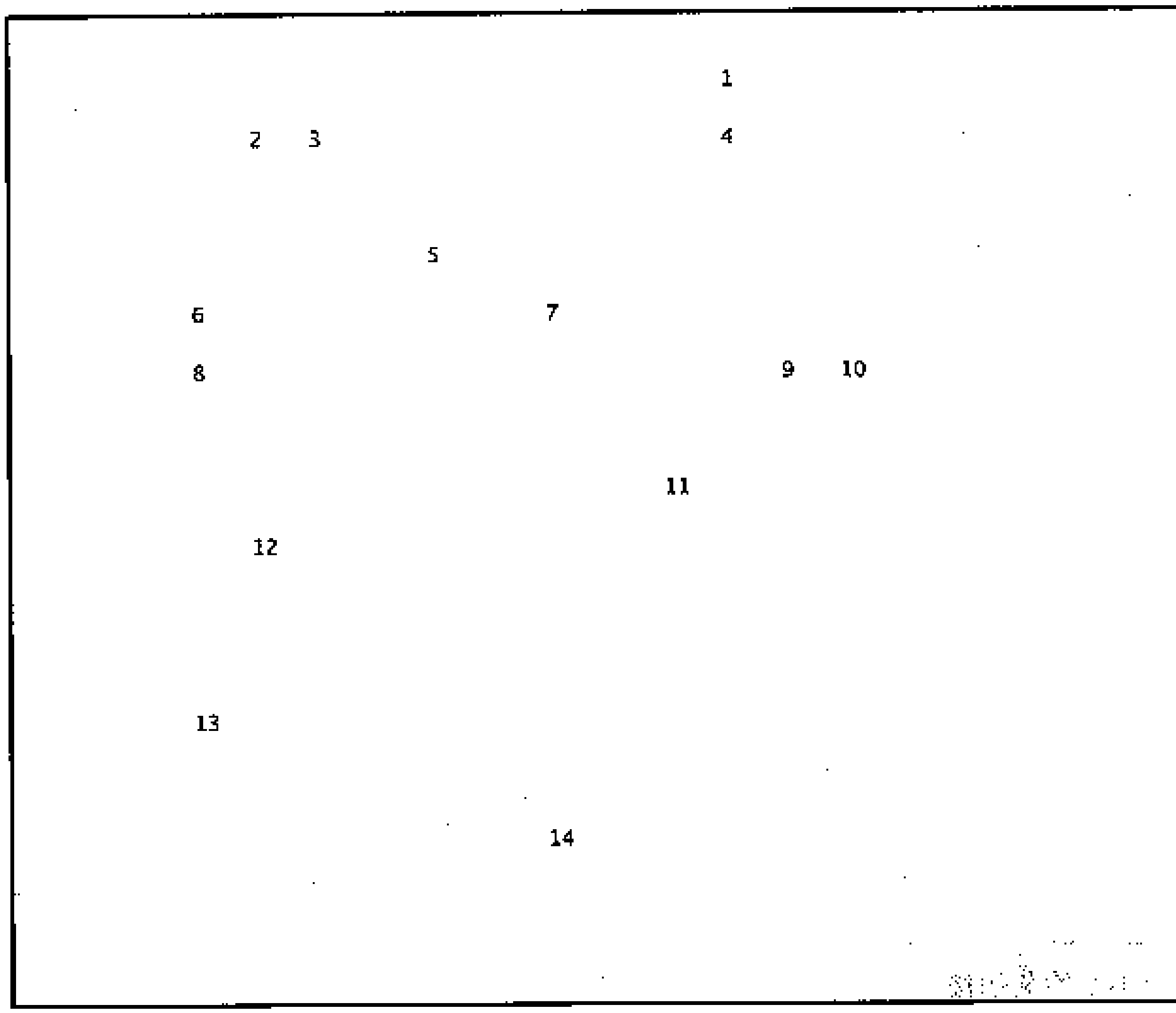
Target属性：对，我总是觉得有些提供信息（额外信息）的窗口会更加便利，而且当人们使用我来打开大量图片时，我特别自豪。这样，用户可以浏览完图片后回到原来的主页。

Head First：OK，你有时也是很便利的。我们会适时地使用你的，但是那些有视力障碍的人是不会过度使用你的。

Target属性：对！

HTML填字游戏

这里是些训练你大脑左半球的急转弯游戏。



横排提示：

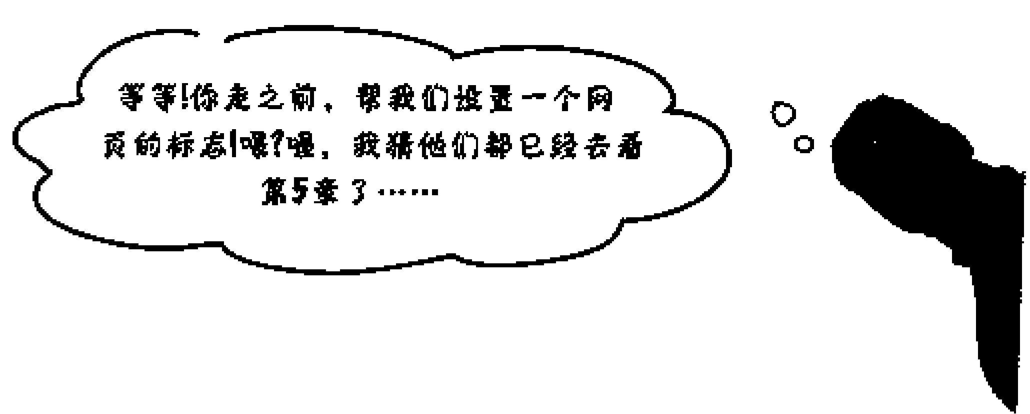
2. URL的错误发音。
4. 为网页中的<a>元素添加锚的属性。
7. Earl卖的东西。
8. 资源的网址。
9. 我们在本章一直使用的协议。
11. 网站独一无二的名字。
12. 通常使用这类链接来链接在同一个服务器中的页面。
13. 请求/响应协议。
14. Mac FTP最常用的应用程序。

竖排提示：

1. 人们用它们代替阅读文字。
3. 从根目录开始的路径。
5. 希望你从Web镇寄回什么？
6. 提供caffeine相关信息的网站。
7. 保持你的链接标签____。
10. 控制域名。
11. 当你请求一个目录时，你会得到一个____文件。
12. 你的网站的最顶层目录。

要点

- 通常将网页发布到网上的最佳途径是找一家主机代理商。
- 域名是一个独一无二的名字，像amazon.com或者starbuzzcoffee.com，都是用来标识网站的。
- 主机代理商可以为你的域创建一个或者多个Web服务器。服务器名字通常是“www”。
- FTP（File Transfer Protocol，文件传输协议）是一种把网页和内容传送到服务器上的常用方法。
- FTP应用程序，例如用于Mac的Fetch或者用于Windows的WS_FTP，它们提供图形用户界面让FTP的使用更加简单。
- URL代表Uniform Resource Locator(统一资源定位符)，或者叫网址，用于标识网络上的资源。
- 通常URL由一个协议、一个网站名和到资源的绝对路径组成。
- HTTP是一种用于在Web服务器和浏览器之间传送网页的请求/响应协议。
- 浏览器用文件协议从你的电脑中读取网页。
- 绝对路径是从根目录到文件的路径。
- “index.html”和“default.htm”都是默认页面。如果你只指定了目录而没有文件名，Web服务器会寻找默认页面返回给浏览器。
- 可以在<a>元素的href属性中使用相对路径或者URL链接到其他网页。要链接到自己的网站的其他页面，最好还是使用相对路径；对外部链接则使用URL。
- 使用id属性在页面中创建目标锚。在“#”后边跟目标锚id，用以链接到页面的特定位置。
- 要提高可读性，使用title属性来提供对<a>元素中的链接的描述。
- 使用target属性在新窗口中打开链接。同时，target属性可能会给使用不同设备和浏览器的用户带来问题。

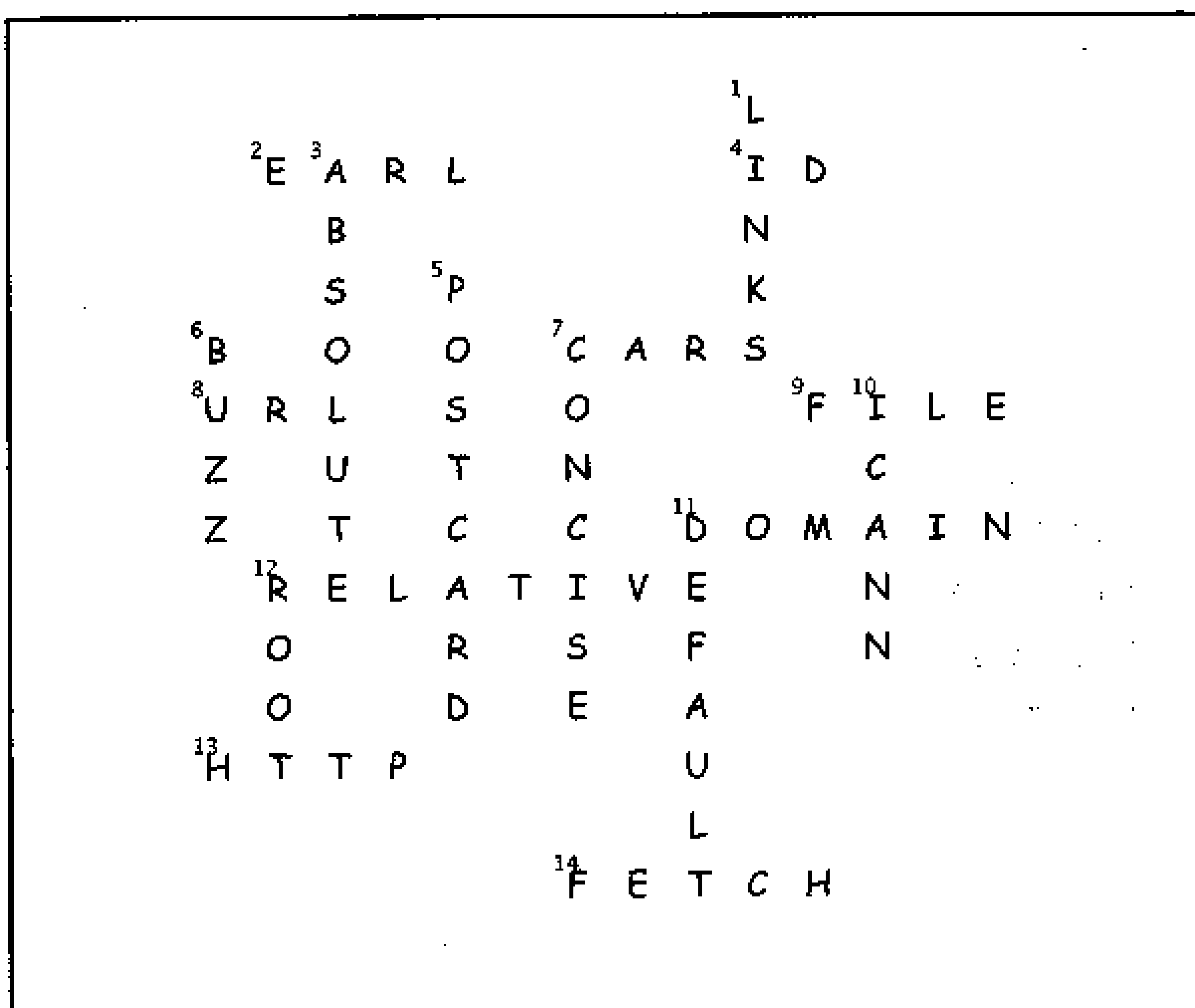


Sharpen your pencil 答案

http ://**www.starbuzzcoffee.com** **/index.html**

协议　　网站名　　绝对路径

你的网站名写在这里。



Earl 的URL需要你的帮助

答案

A



B



C



D



E



练习答案

打开你的Starbuzz中的“index.html”文件并给到“mission.html”的链接添加标题，内容为“Read more about Starbuzz Coffee’s important mission”。注意我们还没有简写这个标签。把链接标签简写为“our Mission”。这里是答案。测试你的修改了吗？

```
<html>
    <head>
        <title>Starbuzz Coffee</title>
        <style type="text/css">
            body {
                background-color: #d2b48c;
                margin-left: 20%;
                margin-right: 20%;
                border: 1px dotted gray;
                padding: 10px 10px 10px 10px;
                font-family: sans-serif;
            }
        </style>
    </head>

    <body>
        <h1>Starbuzz Coffee Beverages</h1>
        <h2>House Blend, $1.49</h2>
        <p>A smooth, mild blend of coffees from Mexico,
            Bolivia and Guatemala.</p>

        <h2>Mocha Cafe Latte, $2.35</h2>
        <p>Espresso, steamed milk and chocolate syrup.</p>

        <h2>Cappuccino, $1.89</h2>
        <p>A mixture of espresso, steamed milk and foam.</p>

        <h2>Chai Tea, $1.85</h2>
        <p>A spicy drink made with black tea, spices,
            milk and honey.
        </p>
        <p>
            Read about <a href="mission.html"
    title="Read more about Starbuzz Coffee's important mission">our Mission</a>
            <br>
            Read the <a href="http://buzz.headfirstlabs.com"
                title="Read all about caffeine on the Buzz">Caffeine Buzz</a>
        </p>
    </body>
</html>
```

添加title属性到mission链接。

移动“Read about”到<a>元素外。

认识媒体

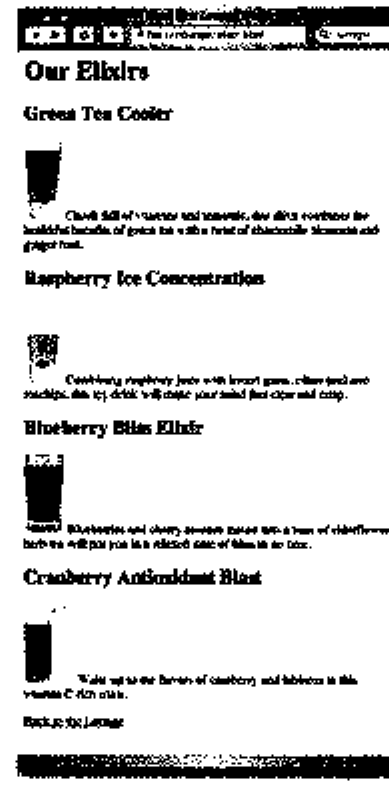


微笑着说“茄子”。实际上，应该是微笑着说“gif”、“jpg”或“png”——当为网页“显影图像”时，你将会用到这些。在这一章，你将学会在网页中添加第一种媒体类型：图像。你想在网上获得你想要的数字图像吗？没问题。想为你的网页添加一个logo吗？没问题。但这之前，我们要再次介绍<img>元素。真是抱歉，并非我们粗心大意，而是我们从来都没“正式地介绍”。为了补偿这一点，我们会用整整一章来介绍它。学完本章后，你会知道如何使用<img>元素和它的属性等所有细节。你同样会看到这个小元素是怎样让浏览器做额外的工作来接收并显示图像的。

浏览器如何处理图像

浏览器处理**<img>**元素的方式和处理其他元素不同。以**<h1>**或者**<p>**为例。当浏览器在页面中遇到这些标签，它只需显示它们，非常简单。但是当浏览器遇到一个**<img>**元素时，就要进行特殊处理：浏览器必须先接收图像，然后才能在该页面上显示图像。

理解上述的最好方法就是看例子。我们快速看一下Head First休闲室中的饮料页面，里面有四个**<img>**元素。




```
<html>
  <head>
    <title>Head First Lounge Elixirs</title>
  </head>
  <body>
    <h1>Our Elixirs</h1>

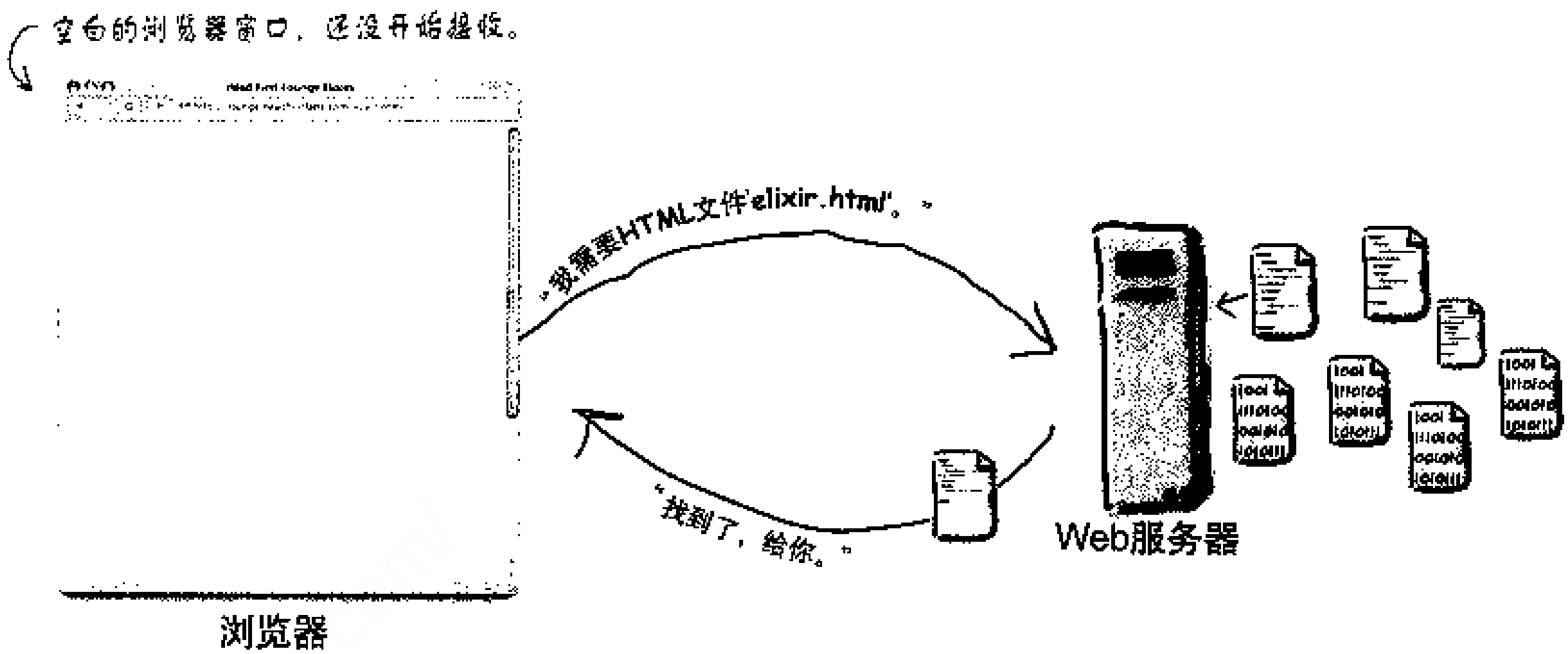
    <h2>Green Tea Cooler</h2>
    <p>
            <img src="../images/green.jpg">
            Chock full of vitamins and minerals, this elixir
            combines the healthful benefits of green tea with
            a twist of chamomile blossoms and ginger root.
    </p>
    <h2>Raspberry Ice Concentration</h2>
    <p>
            <img src="../images/lightblue.jpg">
            Combining raspberry juice with lemon grass,
            citrus peel and rosehips, this icy drink
            will make your mind feel clear and crisp.
    </p>
    <h2>Blueberry Bliss Elixir</h2>
    <p>
            <img src="../images/blue.jpg">
            Blueberries and cherry essence mixed into a base
            of elderflower herb tea will put you in a relaxed
            state of bliss in no time.
    </p>
    <h2>Cranberry Antioxidant Blast</h2>
    <p>
            <img src="../images/red.jpg">
            Wake up to the flavors of cranberry and hibiscus
            in this vitamin C rich elixir.
    </p>
    <p>
      <a href="../lounge.html">Back to the Lounge</a>
    </p>
  </body>
</html>
```

这个HTML中包括四个图像。

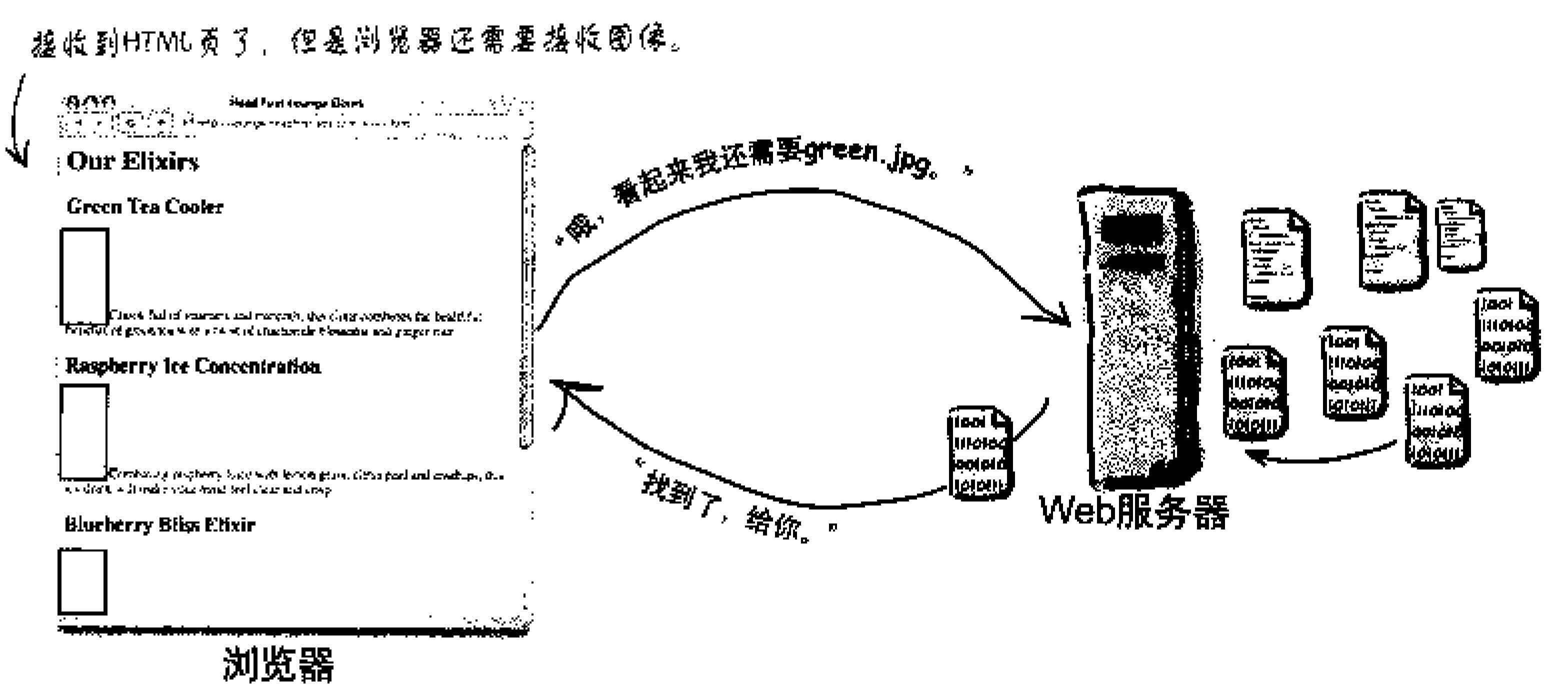
现在让我们分步看看当收到来自http://lounge.headfirstlabs.com的请求时，浏览器怎样接收并显示网页。

幕后花絮

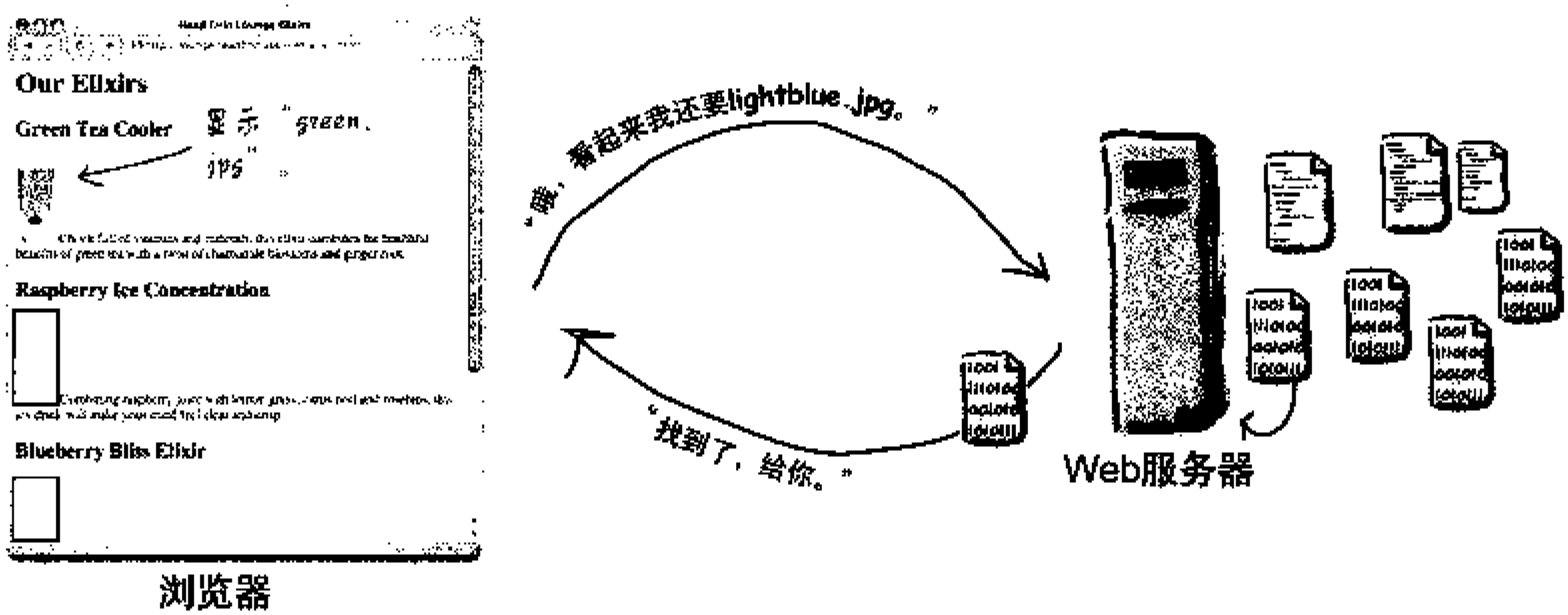
❶ **首先，浏览器从服务器那里接收“elixir.html”文件。**



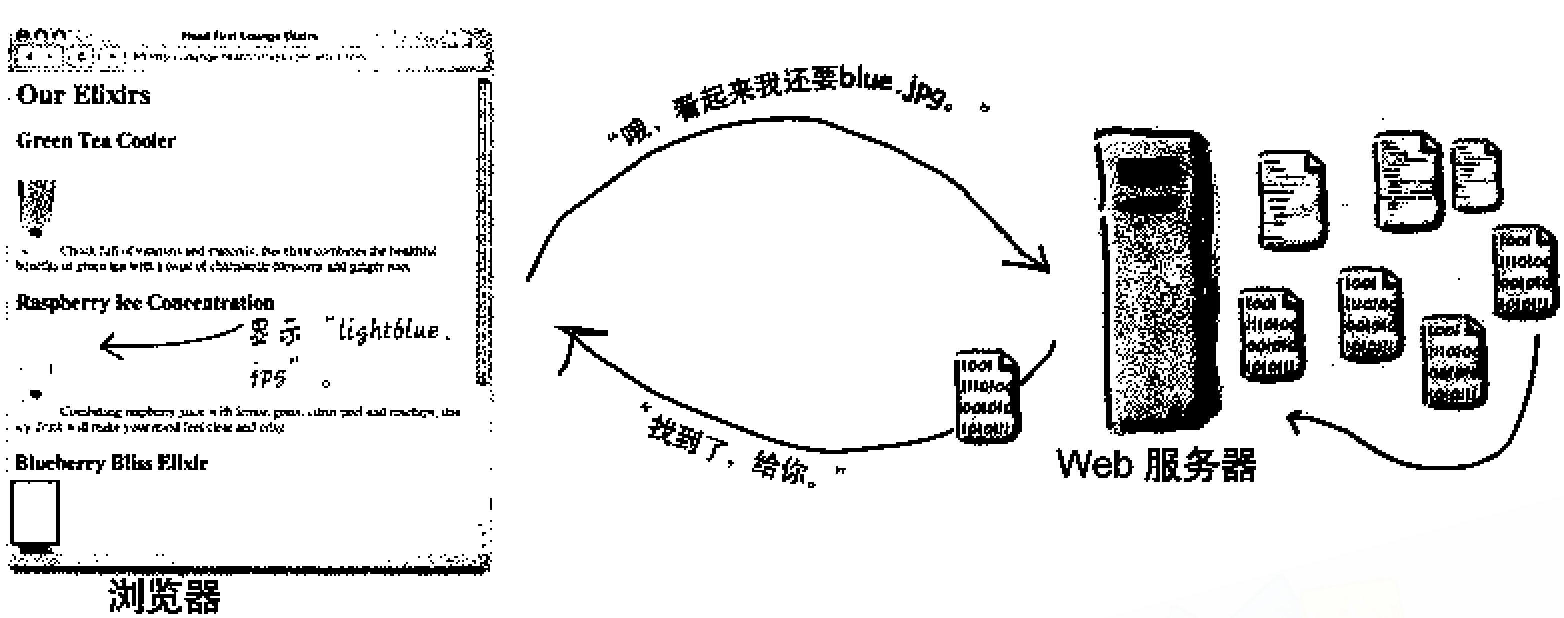
❷ **然后，浏览器阅读“elixir.html”文件，显示、发现要接收的四个图像。所以，它需要从Web服务器接收每个图像文件，从“green.jpg”开始。**



3 接收到“green. jpg”以后，浏览器显示它并开始接收下一个图像：“lightblue. jpg”。



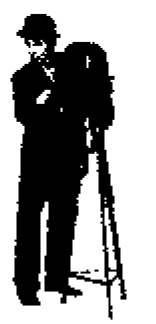
4 现在浏览器接收了“lightblue.jpg”文件，显示图像并开始接收下一个：“blue.jpg”。页面中的每个图像都是这样显示的。



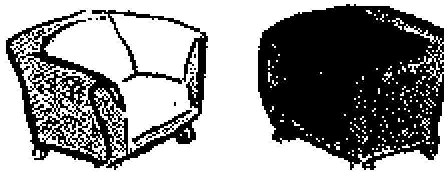
图像是如何工作的

图像就是image，对不对？嗯，其实世界上有很多种图像格式，都有各自的优缺点。但幸运的是，网站中常用的只有两种：JPEG和GIF。什么时候用哪种格式是要讲究技巧的。

JPEG和GIF有什么不同呢？

 照片和复杂图像使用JPEG格式	 纯色图像、logo和几何图形使用GIF格式
可在连续调次（复制品中有中间层次，如照片）图像中获得最好效果。	对于几种纯色组成的图像、线条组成的图像（如logo、剪贴画）和含有小段文字的图像，使用GIF比较合适。
可以用1600万种不同的颜色显示图像。	用256种颜色显示图像。
是一种“有损”格式，因为文件缩小时会丢失部分图像信息。	GIF同样会压缩文件来减小尺寸，但是不会丢失任何东西，是种“无损”格式。
不支持透明。	允许把背景颜色设为“透明”的，图像背景就可以穿透显示。

围炉夜话



今夜话题：JPEG和GIF比较它们的图像。

JPEG

你好，GIF。我们刚在网页上见过面吧？

GIF

嗯……如果每个人都坚持使用GIF该多好！那样我就不必经常碰到你了。

JPEG

哈哈，如果你能很好地显示复杂图像，如照片的话，我肯定人们会很乐意坚持使用你的，但是你始终不懂得怎么显示那些超过256种颜色的东西。

GIF

嘿，如果像你那样不顾质量地显示照片也是很简单的。但我是完全为质量着想的。如果我不能完全显示一张图像，我就不会去做。瞧瞧你试图去显示的那些logo吧……讨厌。

JPEG

你想跟我谈质量？我让我的用户选择他们想要的精确质量。

GIF

对，但是以什么为代价的呢？承认吧，为了获得能在网页上传送的大小合理的图像，你必须损失一些图像质量。

JPEG

的确如此，但是人们乐意这样做。不是每个人的页面上都需要超高分辨率的图像。因为有我，用户可以自由设定质量为低或中，来获得比较满意的图像质量。但如果他们使用你，就需要更大的文件来存储同样的图像。

GIF

是，但是你注意过线条，logo，小文本，单色图像吗？用JPEG似乎不见得怎么样。

JPEG

没错，GIF能够很好地处理那些，但前提是他们的颜色种类比较少。你看起来就像是我的低一级版本。我可以做你能做的任何事情。

怎么回事？GIF？我们在做节目呢？你去哪里？

我想你有点过分依赖透明这个东西了——我指的是在图像上添加背景颜色。

我还不太担心这个，绝大多数照片是有背景的。

怎么会发生这种事呢？

那好，坚持你的logo和简单文本图像，而我就坚持于照片和复杂图像。每个人都知道我比较善于处理复杂的图像。

GIF

{GIF突然消失了}

{GIF重新出现}

别慌。我只想证明一个观点。如果JPEG那么伟大的话，你的图像为什么不能像我的这么透明呢？使用透明，图像背景就会穿透显示。如果我的用户想在网页上添加一个logo，而且页面背景是彩色的，他们会使用我，因为他们知道我会让背景穿透logo显示而不带任何颜色。

当然，这样就可以改变网页的颜色了。没有别的出路。透明是唯一的方法。要实现它，你必须使用我。

哦，真的？那如果你想剪切出一张人物或者一棵树的图像粘贴在网页上又不想要背景，那怎么办？

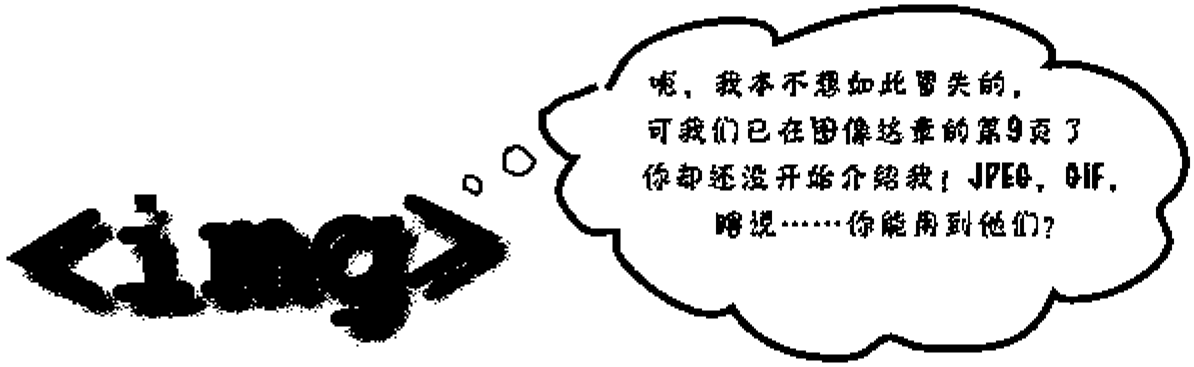
你会惊讶于我在显示图像方面被使用的频繁性，因为我的用户想要透明背景。

嘿，有人要我去做透明图像了……要走喽。

用哪种图像格式？

祝贺你：你当选为今天的“盛大图像格式选择者”。给每张图像选择网上的最佳显示格式。

	JPEG	或者	GIF
	☐		☐
	☐		☐
	☐		☐
	☐		☐
	☐		☐



现在正式介绍：认识<img>元素。

我们已经把介绍拖延得够久了。正如你所见的，图像要处理的事务比HTML标记还多。无论如何，现在已经足够了……是时候来认识**<img>**元素了。

我们还是先来仔细观察一下这个元素（尽管你现在可能已经知道**<img>**元素的大多数工作原理）。

这是<img>元素。

<img>元素是个内联元素，前后不会插入换行。

```
<img src="images/drinks.gif">
```

Src属性指定了要在网页上显示的图像文件的位置。

你已经知道<img>元素是个空元素。

这就是**<img>**？不完全。还有许多你要知道的属性呢。当然你同样想知道如何使用**<img>**元素来引用网上的图像而不只是你自己网站里的。但你其实已经了解了使用**<img>**元素的基本内容。

让我们来学习使用**<img>**元素的几个重点，然后学以致用。

<img>：不再是相对链接

src属性不只用于相对路径，也可以把URL放到src属性中。存储于网络服务器的图像会伴随在HTML页面左右，所以网上的每张图像都有自己的URL，就像网页一样。

如果想指向另外一个网站的图像，你通常会用URL定位它（记住，如果链接和图像都在相同的网站，用相对路径更好）。

这里教你如何用URL链接到图像：

```
<img src="http://www.starbuzzcoffee.com/images/corporate/ceo.jpg">
```

使用URL指定图像，只需把整个图像的URL放到src属性中。

URL是到图像的路径，所以结尾的文件名通常是图像文件名。对网页来说，没有什么所谓的默认图像。

Sharpen your pencil

这是“Sharpen your pencil”，和铅笔有关（哦，同样还有图像）。这个练习包括一些问题：给你一根普通的新铅笔，如果你想用它来画一道实线，用一整支铅笔，线能画多长？

这和图像有什么关系？你必须写些HTML来找到这个答案。这些问题的答案包含在这个URL：http：//www.headfirstlabs.com/trivia/pencil.gif下的图像中。你的工作是给HTML添加一张图像并找到答案。

```
<html>
    <head>
        <title>Sharpen your pencil trivia</title>
    </head>
    <body>
        <p>How long a line can you draw with the typical pencil?</p>
        <p>

        </p>
    </body>
</html>
```

把你的image元素放在这里。

there are no Dumb Questions

问：<img>元素很简单——就是提供一种途径来指定需要在页面中显示的图像的位置？

答： 对，仅此而已。我们将介绍许多给元素添加的属性。以后，你会看到许多使用CSS来改变图像的视觉样式的方法。

但关于图像还有许多要知道的。不同的图像格式有什么意义？图像什么情况下能替换？它们应该有多大？我该怎么为网页准备图像？

问：我们已经知道空元素就是没有内容的元素。我们在学习<img>时知道它也是空的。但是，它不是有内容（图像）吗？

答： 哦，准确来说，空元素是HTML页中开始标记和结束标记之间没有任何内容的元素。图像是内容，但是<img>元素是关于图像的，而图像不是HTML页的部分。当浏览器显示页面的时候，图像代替了<img>元素。记住，HTML页是纯文本，所以图像永远不是页面的一部分。它是独立的。

问：回到网页加载图像的例子……当我加载一个网页时，我看不到图像一个接一个地加载。为什么啊？

答： 浏览器经常是同时接收图像的。就是说，浏览器同一时间请求多个图像。视电脑和网络的速度而定，足够快的话，通常你会看到页面和图像一起显示。

问：如果我看到网页上的一张图像，我怎么确定它的URL以便链接它？

答： 大多数浏览器允许你"右击"图像，这样会弹出一个有选项的菜单。在选项中你会看到"复制图像地址"或者"复制图像链接"，这会把URL放到你的粘贴板上。另外一个找到URL的方法是选择"在新窗口打开图像"，这会在浏览器新窗口打开图像。然后你就可以从浏览器的地址栏中得到图像的URL。最后一个选择是使用浏览器的"查看源文件"菜单选项并通读HTML。注意，你可能找到一个图像的相对链接，所以你必须用网站的域名和图像的路径"重构"URL。

问：JPEG和GIF孰优孰劣？

答： "优于"通常是结合图像质量和文件大小而言的。一个JPEG照片通常比同质量的GIF小，而GIF logo通常看起来更佳，而且文件比JPEG格式的小。

问：我还听说过PNG图像格式。为什么你们没有提到它呢？

答： PNG是最新型的图像格式，是个有趣的格式，支持JPEG和GIF图像样式。而且比GIF有更高级的透明特点。现在，PNG使用不太广泛，因为不是所有浏览器都支持它。但是它的普及性在迅速增长。你可以随意使用PNG，但是要小心，它不能在所有浏览器上使用。

总是提供可选项

毫无疑问，你无法准确知道人们浏览网页时用的是什么浏览器和设备。访问者可能会用移动设备，为视觉有障碍的用户设计的屏幕读取器，网络连接很慢的浏览器（网站可能只能接收到文本却看不到图像），移动电话，能上网的烤面包机……谁知道呢？

但是在这些不确定的设备中你可以做好准备。即使浏览器不能显示你页面中的图像，还有其他选择。你可以使用<img>元素中的alt属性来为访问者提供图像信息。以下是它应用的示范：

```
<img src="http://www.headfirstlabs.com/trivia/pencil.gif"
     alt="Pencil line 35 miles long">
```

这个alt属性要求是一小段描述图像的文字。

如果图像不能显示，就用这段文字代替。

练习

在这个练习中你会看到当碰到损坏图像时，浏览器是如何使用alt属性的。理论上如果一个图像无法显示，alt属性将会代替它显示。但不是所有浏览器都能实现的，所以你的结果也许会不同。这里是你所要做的：

1. 从上个练习中得到你的HTML。
2. 加上alt属性“pencil line 35 miles long”更新图像元素。
3. 把图像名由“pencil.gif”改为“broken.gif”。这个图像并不存在，所以你会得到一个损坏的图像。
4. 用浏览器重载页面。
5. 最后，下载其他浏览器试一试。你有没有得到不同的结果？

举例来说，你可以试试Firefox(http://www.mozilla.org/)或者Opera(http://www.opera.com/)。

查看章后我们的参考答案……

调整图像尺寸

<img>元素还有一个属性你应该了解，事实上它们是一对属性：**width**（宽度）和**height**（高度）。你可以使用这两个属性预先告诉浏览器页面中图像的大小。

这是使用**width**和**height**的例子：

```
<img src="images/drinks.gif" width="48" height="100">
```

Width属性告诉浏览器页面中图像应该显示的宽度。

Height属性告诉浏览器页面中图像应该显示的高度。

高度和宽度都是用像素的数量指定的。如果你不熟悉像素，我们稍后会介绍。你可以给任何图像添加width和height属性；如果不添加，浏览器会在页面显示前自动决定图像的大小。

there are no Dumb Questions

问：如果浏览器能计算出图像大小，为什么我还要使用这些属性？

答： 如果你在HTML中提供了宽度和高度，许多浏览器会在显示网页之前先安排好布局。如果没有提供，浏览器通常会在它知道图像大小之后重新调整网页布局。记住，浏览器是在下载完HTML文件并显示页面后才下载图像。除非你告诉它，否则浏览器在下载之前是不知道图像大小的。

你同样可以提供比图像尺寸更大或更小的宽度和高度值，浏览器会按比例缩放图像来适应这些尺寸。当需要比原来尺寸更大或更小地显示一个现有的图像时，许多人会这么做。就如你以后会看到的一样，然而，这种用法不值得提倡。

问：我必须成对地使用这些元素吗？我可以单独指定高度或者宽度吗？

答： 可以，不过既然你不怕麻烦告诉了浏览器一个尺寸，提供第二个尺寸也仅是相同的工作量而已(提供单是宽度或者高度不会获得什么便利，除非你要把图像缩放到特殊的高度和宽度)。

问：我们说过许多次，HTML是用于结构的，而不是外观。这些好像是外观属性。是吗？

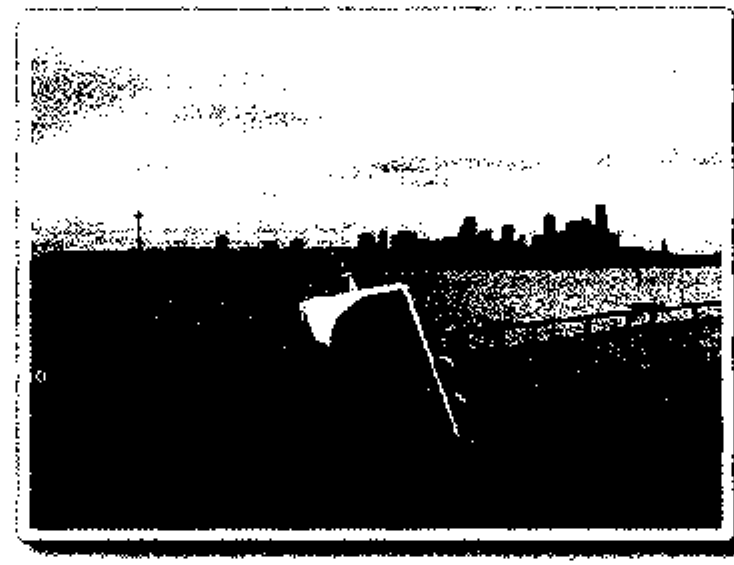
答： 这取决于你如何使用这些属性。如果你把图像的宽度和高度设定为原来的尺度，那他们就是提供信息的。然而，如果你使用width和height来调整浏览器中的图像大小，那你是在为外观使用属性。在这种情况下，可能用CSS来达到这个目的比较好。

创建最后的fan网站：myPod

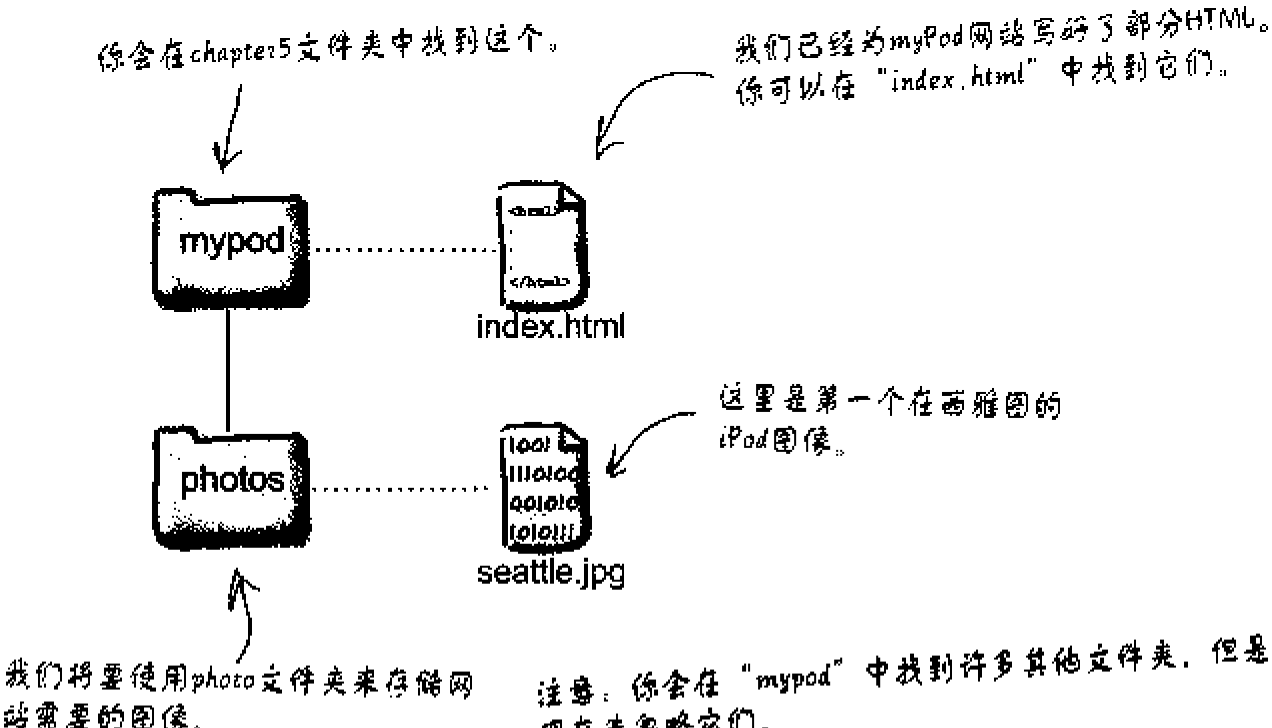
拥有iPod的人都喜爱他们的iPod，他们无论到哪里都带着iPod。想象创建一个名为“myPod”的新网站来显示你朋友和他们喜欢的世界各地的iPod图像。

你要怎样开始？只要些有关HTML的知识，一些图像，和你对iPod的喜爱。

我们已经为网站写好了部分HTML，但是还没有添加图像——这就是你的工作。在你开始处理这些图像前，我们先设置一些东西；看看书中的实例资源中的“chapter5”文件夹。你会找到一个叫“mypod”的文件夹。打开“mypod”文件夹，你会看到：



我的iPod在西雅图！你可以看到乌云和太空针，但看不到628咖啡屋。



检查mypod的“index.html”文件

打开“index.htm”文件，你会看到myPod已经在工作了。
下面是现在的HTML。

```
<html>
  <head>
    <title>myPod</title>
    <style type="text/css">
      body { background-color: #eaf3da;}
    </style>
  </head>
  <body>

    <h1>Welcome to myPod</h1>
    <p>
      Welcome to the place to show off your iPod, wherever you might be.
      Wanna join the fun? All you need is any iPod from the early classic
      iPod to the latest iPod Nano, the smallest iPod Shuffle to the largest
      iPod Photo, and a digital camera.  Just take a snapshot of your iPod in
      your favorite location and we'll be glad to post it on myPod. So, what
      are you waiting for?
    </p>

    <h2>Seattle, Washington</h2>
    <p>
      Me and my iPod in Seattle!  You can see rain clouds and the
      Space Needle.  You can't see the 628 coffee shops.
    </p>

  </body>
</html>
```



我们把已经写好的CSS放到这里。现在把它们输进去——它的作用是给网页添加淡绿色的背景。我们保证会在后述的章节学习CSS!

这个HTML看起来很熟悉，因为我们使用了基本构建模块<h1>、<h2>和<p>。

这是它在浏览器中的样子。不是很糟糕，但我们需要图像。



Sharpen your pencil

正如你所见的，myPod的大部分HTML已经写好并准备运行了。你所要做的是为每张你想加入的图像添加一个**<img>**元素。现在这里有一个图像："seattle.jpg"，开始添加一个元素使图像显示在页面的下方。完成后，用浏览器加载页面并查看Seattle（西雅图）的风景。



这里是你需要放置第一张照片的地方。

```
<html>
   <head>
      <title>myPod</title>
      <style type="text/css">
         body { background-color: #eaf3da;}
      </style>
   </head>
   <body>

      <h1>Welcome to myPod</h1>
      <p>
         Welcome to the place to show off your iPod, wherever you might be.
         Wanna join the fun? All you need is any iPod from, the early classic
         iPod to the latest iPod Nano, the smallest iPod Shuffle to the largest
         iPod Photo, and a digital camera.  Just take a snapshot of your iPod in
         your favorite location and we'll be glad to post it on myPod. So, what
         are you waiting for?
      </p>

      <h2>Seattle, Washington</h2>
      <p>
         Me and my iPod in Seattle!  You can see rain clouds and the
         Space Needle.   You can't see the 628 coffee shops.
      </p>

      <p>

      </p>

   </body>
</html>
```

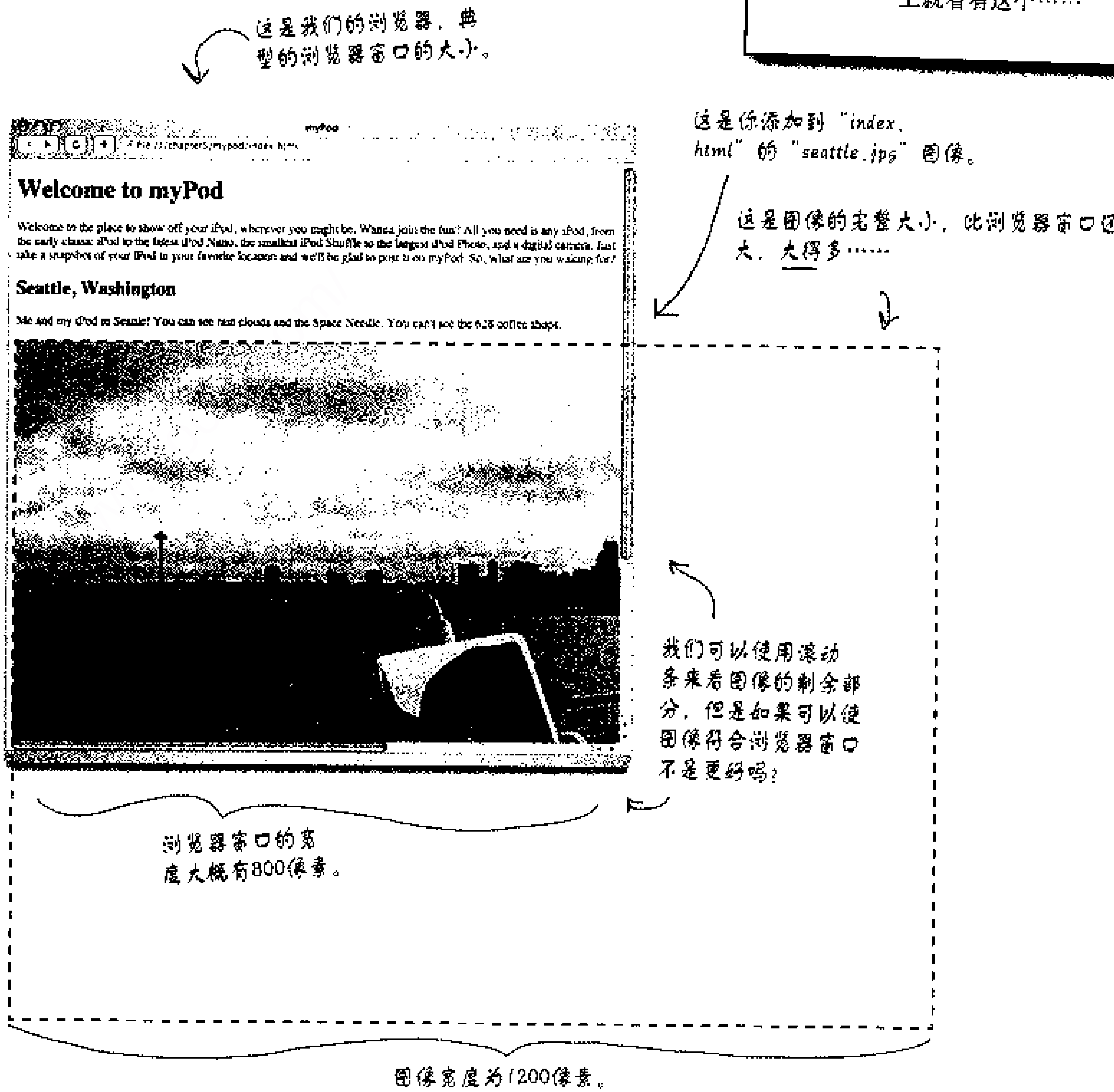
你的<img>元素要写在这里。

哇，图像太大了

呃，现在图像应该就位了，但是它是个大图像。大多数这类来自数码相机的图像都那么大 （或更大)。我们应该让图像保持原样并让浏览者使用滚动条吗？你将会从许多原因中得知这并非是个好主意。

让我们看看图像和浏览器，并看看这种情况是如何的糟糕吧……

> **注意！** 如果图像大小刚好适合你的浏览器窗口，浏览器可能打开了一个“自动调节图像大小”选项。马上就看看这个……



there are no Dumb Questions

问：用户使用滚动条来看图像会有什么问题吗？

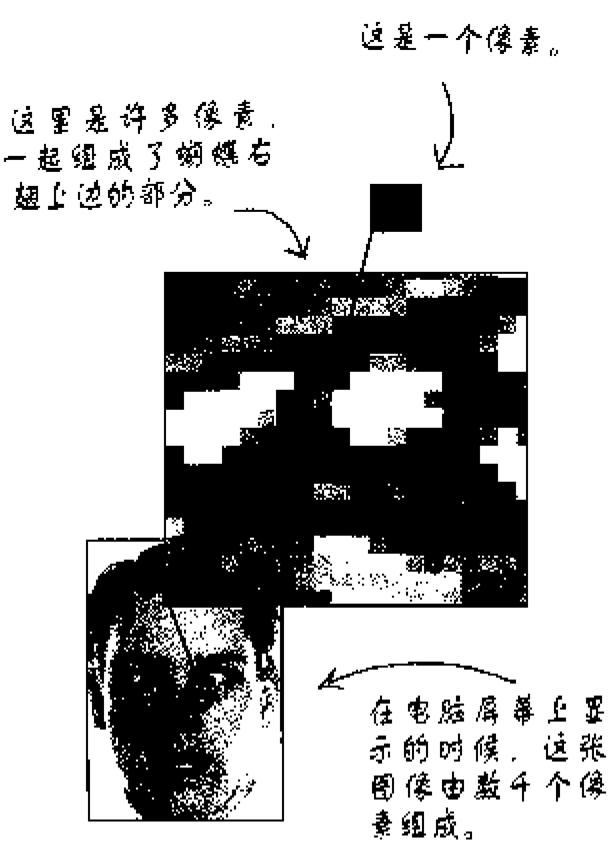
答： 通常来说，加载有大图像的网页较难使用。不但浏览者一次看不到完整的图像，而且使用滚动条很繁琐。大图像需要从服务器传送更多数据到浏览器，会耗费许多时间并导致网页显示速度很慢，尤其是拨号上网或者其他慢速连接的用户。

问：为什么不能使用width和height来调整网页上的图像的尺寸？

答： 因为浏览器在缩小大图像来配合网页之前要先下载整个大图像。

问：你说浏览器窗口的宽度是800像素，具体是什么意思？

答： 你的电脑显示屏是由数以百万计的称为像素的点组成的。贴近看你的显示屏，就会看到它们：



显示屏的尺寸和分辨率有很多种（有些人的显视器小，有些人的大），800像素是大多数人设置的浏览器的典型宽度。所以，800像素作为图像的最大宽度是最佳的显示方案（网页也一样，我们会在稍后的章节中提到）。

问：像素的数目和屏幕上图像的尺寸有什么关系呢？

答： 72像素/英寸是个非常好的方案，视你的显视器而定，你可以提升到120像素/英寸，假设你的显视器是72像素/英寸，如果你想一张宽和高近似3英寸的图像，你要用72(像素)×3(英寸)=246像素，或者四舍五入到250像素。

问：好，那我的图像应该设置为多大啊？

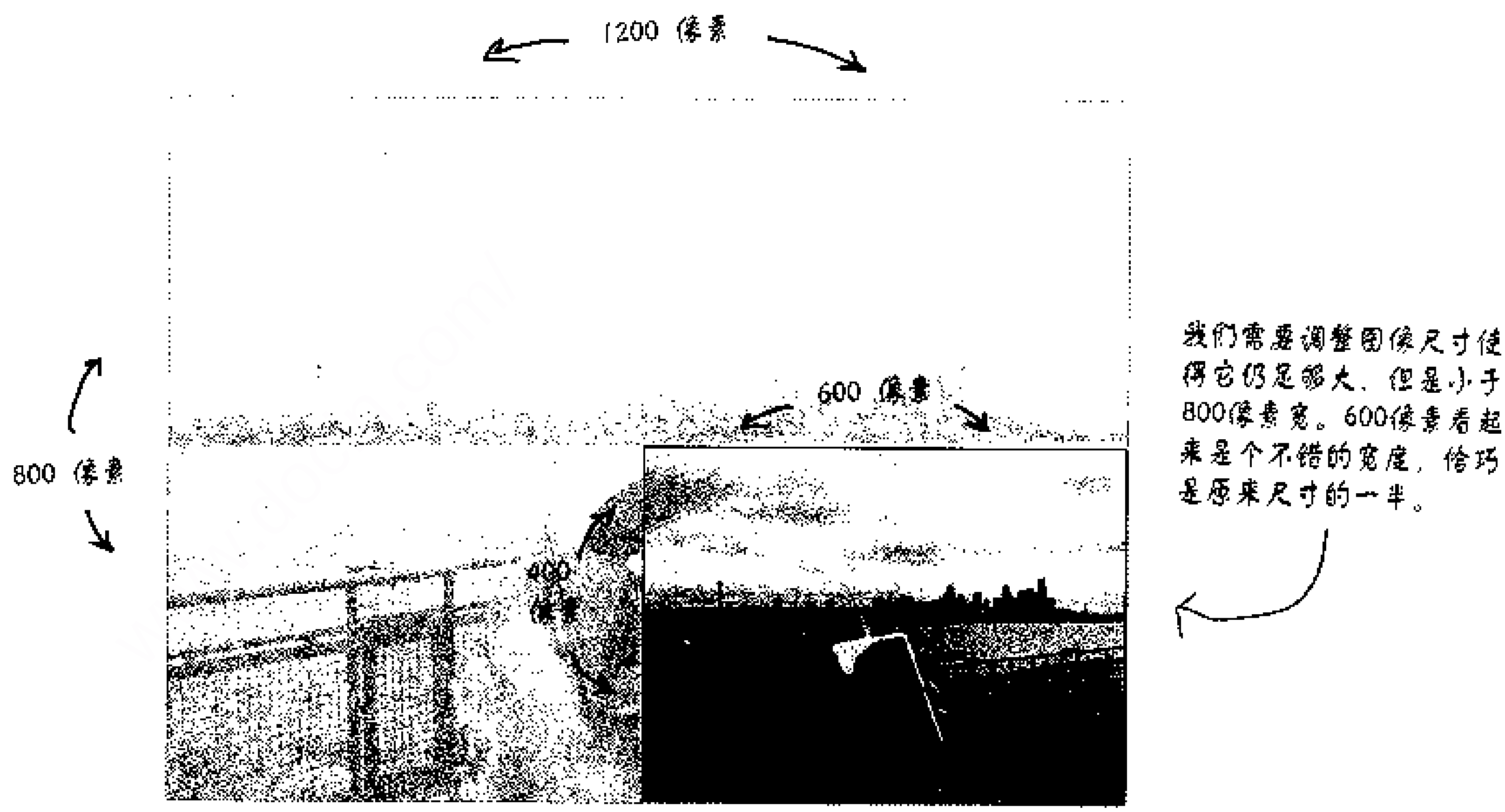
答： 通常来说，你得保持你的图像宽度小于800像素。当然，你可以让它更小，这视图像的用途而定。如果图像是页面中的logo呢？你可能想更小，但是要清晰。毕竟，你不需要一个跟整个网页一样宽的logo。logo宽度往往是在100～200像素之间。所以，归根结底，你问题的答案要视你网页的设计而定。作为照片的话——当然你想尽量大点以便查看——你也许希望网页中的图像小一点，这样加载会比较快，并且允许用户点击每张缩略图来放大观看。我们以后会讲到。

问：我想我的浏览器自动调整了Seattle图像的尺寸，因为它在窗口里看起来非常合适。为什么浏览器会这么做呢？

答： 某些浏览器的特点是调整不符合浏览器窗口规格的图像的大小。但是许多浏览器不会这么做，所以你别想依靠它。即使每个浏览器都有这个特点，服务器和浏览器之间还是要传送比原来所需更多的数据，这会导致页面加载变慢并降低实用性。

调整图像大小以符合浏览器规格

让我们修改图像以符合浏览页规格。现在图像是1200像素×800像素（后面你会看到是如何确定的）。因为我们希望图像宽度小于800像素，我们需调整宽度以适合我们的myPod页。整个myPod的重点是浏览在各地不同环境下的iPod图像，所以我们可能希望图像能足够大。如果我们把图像尺寸缩小到一半，即600像素×400像素，那样会占据浏览器的大部分宽度，两边还能有些空白。不错吧？我们来调整图像大小……

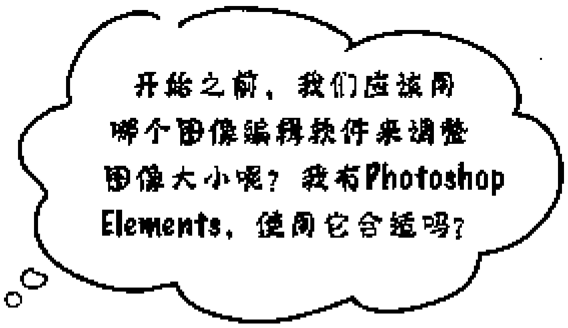


这是你要做的：

❶ 用图像编辑软件打开图像。

❷ 减小图像尺寸到原来的一半(到600像素×400像素)。

❸ 保存图像为“seattle_med.jpg”。





好问题——市场上有许多图像编辑软件（有些是免费的），他们都很类似：我们使用Adobe的Photoshop Elements来调整图像大小，因为它是最流行的图像编辑软件之一，而且在Windows和Macintosh系统中都能运行。如果你有别的图像编辑软件，毫无疑问你可以试试并把每个编辑任务转换到你自己的软件。

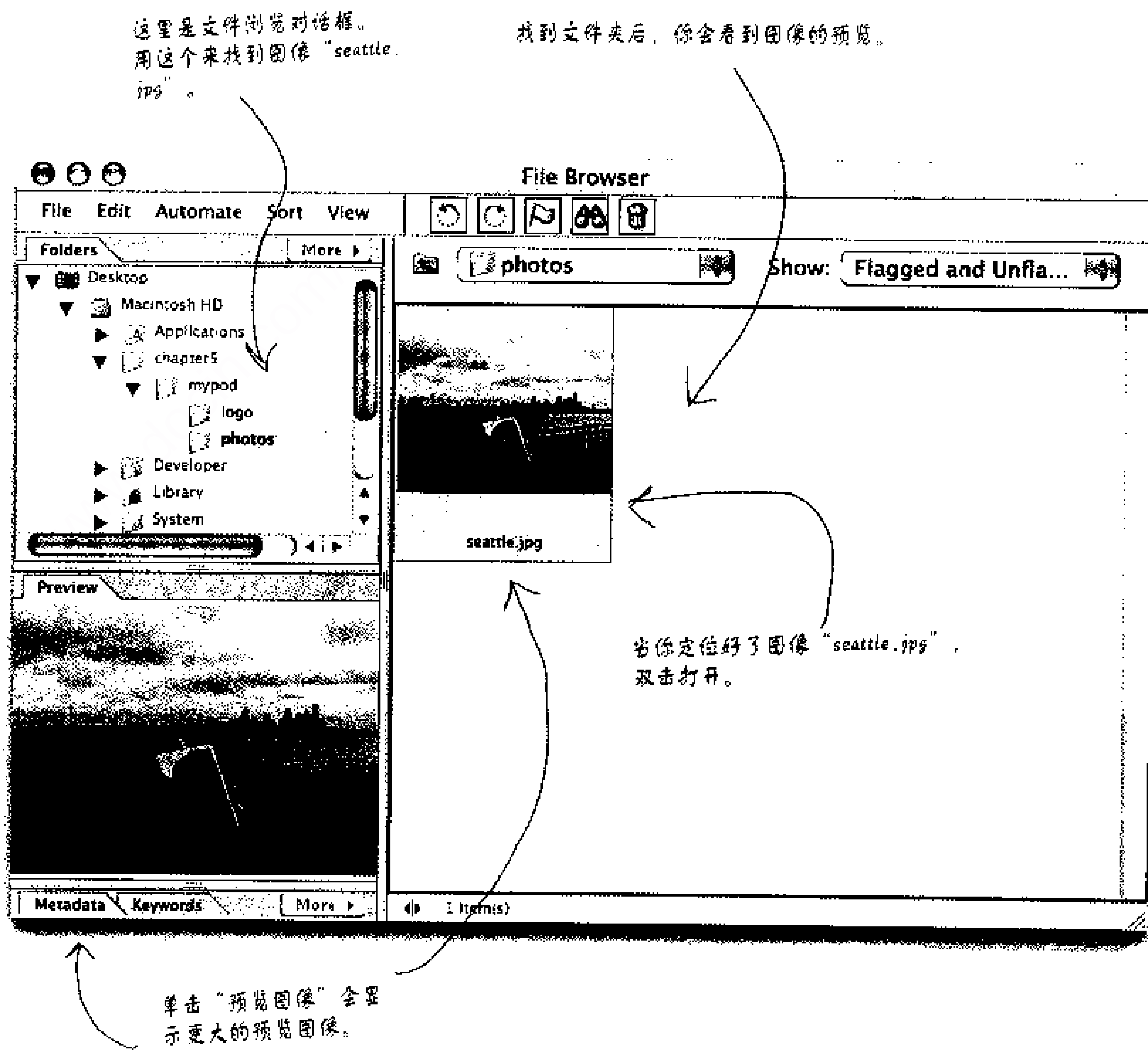
如果你没有图像编辑软件，你可能首先要看看在你的操作系统中是不是自带一个。如果你用的是Mac，你可以使用iPhoto来编辑图像。如果你用的是Windows，你可以找到Microsoft的Digital Image Suite(微软的一个数字相片处理软件)。如果你还没有找到一个编辑软件的话，你可以跟随我们学习如何使用实例文件夹中的HTML和图像来处理该问题。

如果你没有Adobe Photoshop Elements，但是你又想与本书同步，你可以下载并免费试用30天。下载的URL是：http://www.abode.com/products/tryadobe/main.jsp。

打开图像

首先，启动你的图像编辑软件并打开“seattle.jpg”图像。在Phoroshop Elements中，你要选择“文件”菜单下的“浏览文件夹……”选项，打开“文件浏览”对话框。你可以在“chapter5/mypod/photos”文件夹下找到图像“seattle.jpg”。

如果你使用Windows，使用文件下的打开菜单来直接打开图像“seattle.jpg”。

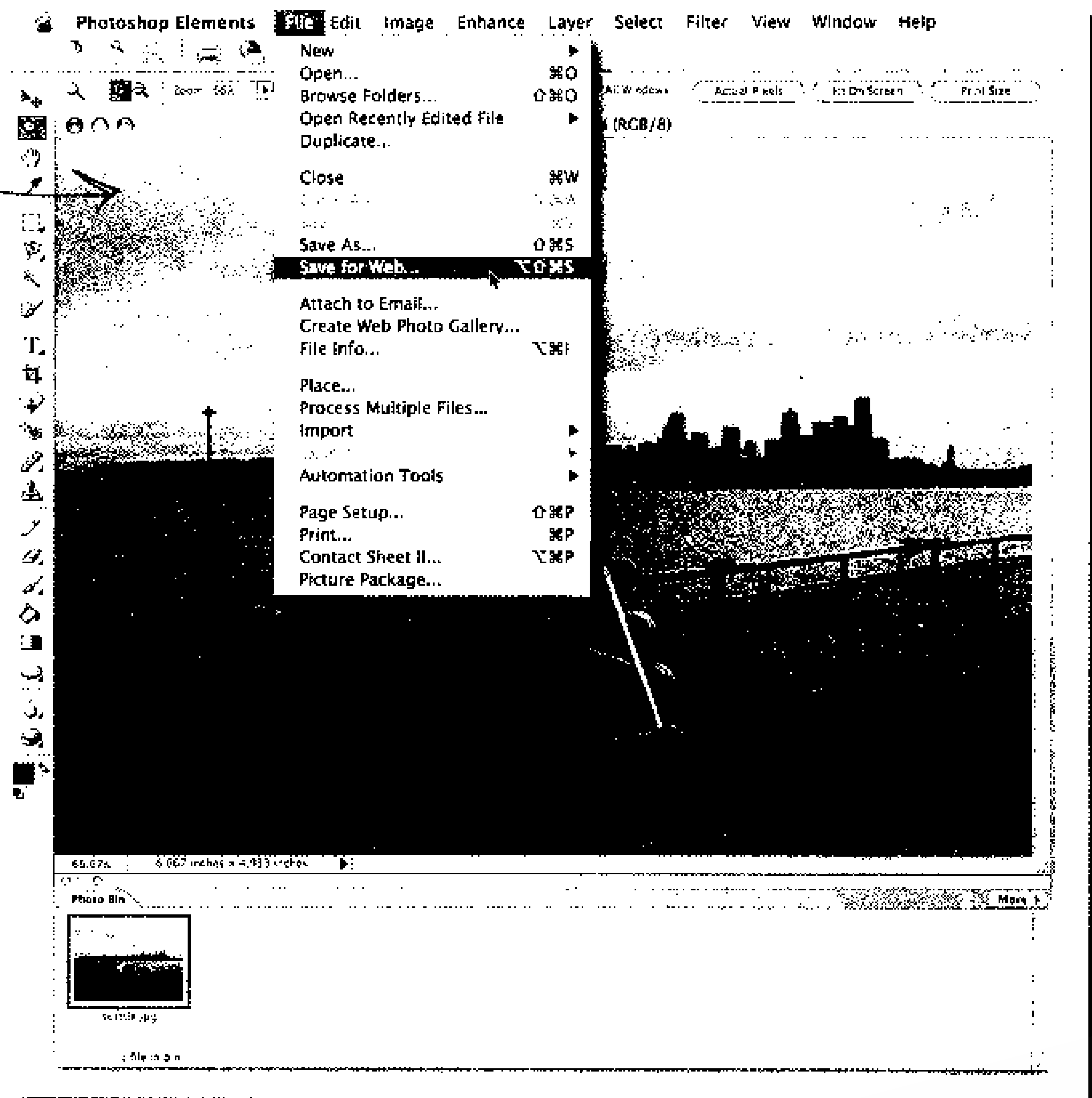


调整图像大小

现在“seattle.jpg”已经打开了，我们要使用“保存为Web所用格式”对话框来调整大小并保存。要来到这个对话框，选择“文件”下的“保存为Web所用格式”。

从文件菜单中选择“保存为Web所用格式”来调整图像大小。

这是在Photoshop Elements下的“seattle.jpg”图像。

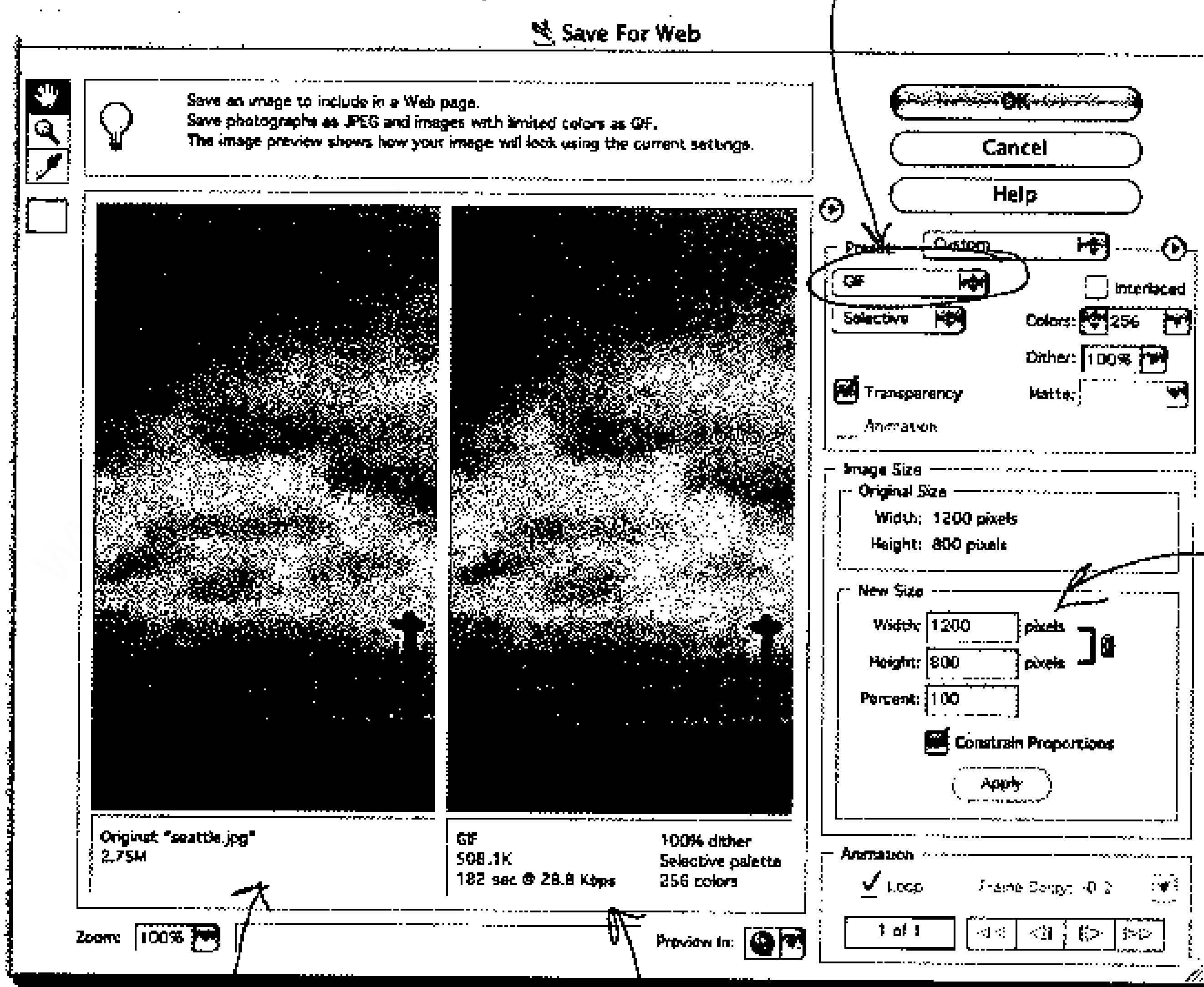


继续调整图像大小……

选择“保存为Web所用格式”这个选项后，你会看到以下的对话框，我们在使用之前先熟悉一下。

这个对话框可以让你做很多有趣的东西。现在，我们要集中注意力于如何用它调整图像大小并保存为网页所需的JPEG格式。

在这里选择保存图像的格式。现在是保存为GIF，随后两页，我们就会改成JPEG……

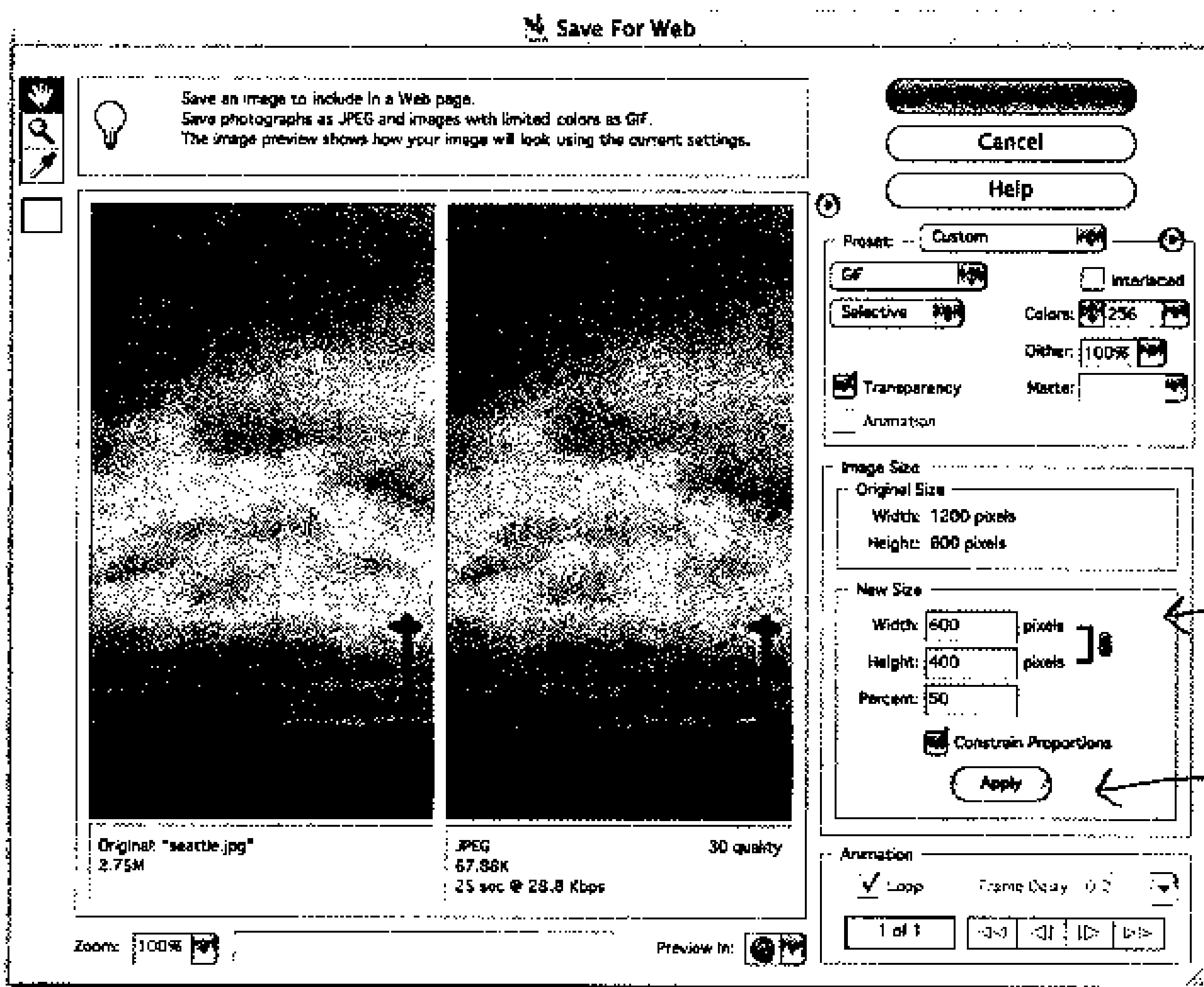


这里是图像当前尺寸：1200像素×800像素。

分离的窗口左边显示的是原来的图像，右边显示的是以你选择的格式保存的图像。当前显示的是GIF格式，我们要在下一步把它改成JPEG格式。

正如你所见，这个对话框中有许多功能，我们可以好好运用它。为了调整图像大小，你需要改变宽度为600像素，高度为400像素。然后你需要用JPEG格式保存图像。我们先从调整大小开始……

(1)改变图像大小，宽度为600，高度为400。如果你选择了"约束比例"，你所要做的就是输入新的宽度600，Elements会自动把高度改成400。



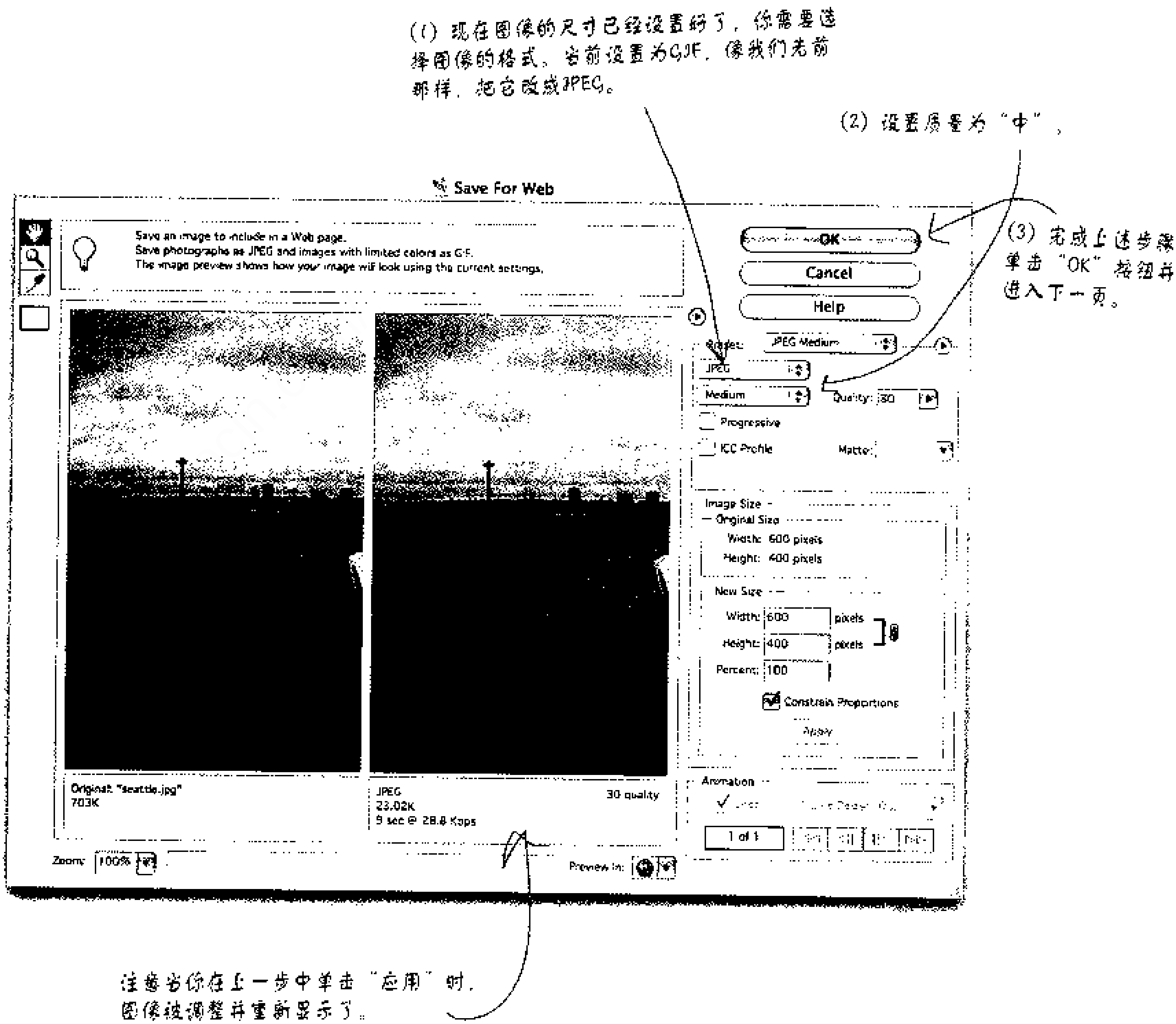
(2)设置好宽度和高度后，单击"应用"按钮来让Element知道这是你想要的尺寸。

这不会影响你原来的图像，只是保存你要的文件。

你要单击"应用"按钮来减小图像尺寸，否则图像将会以原大小保存。

你已经调整完了——那就保存吧

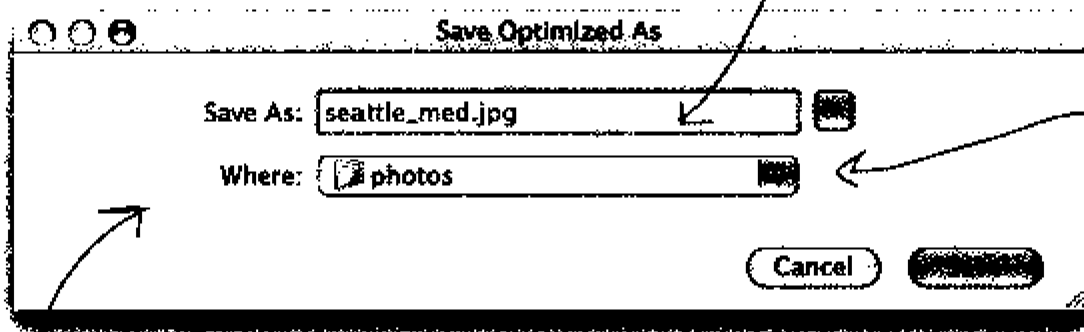
现在你需要以正确的格式(JPEG)来保存图像。你需要选择JPEG格式并设置质量为“中”。我们以后会谈到质量。



保存图像

单击“OK”按钮以后，你会看到保存对话框。保存图像为“seattle_med.jpg”，这样就不会覆盖原来的照片。

更改文件名为seattle_med.jpg。



确保你把图像保存到“mypod/photo”文件夹。

注意你把文件名从“seattle.jpg”改成“seattle_med.jpg”。为什么？人们通常喜欢保存他们原来的高质量大图像来打印，把小的版本放到网上。如果你保存为“seattle.jpg”，原来的照片就会丢失！

单击“save”按钮保存图像。

---

there are no Dumb Questions

问： 你可以多讲一些关于“保存为Web所用格式”中的质量设定的内容吗？

答： JPEG格式允许指定你需要的图像质量等级。质量越低，文件越小。如果你看“保存为Web所用格式”对话框上的预览面板，你会发现当你改变质量等级时，质量和文件大小都会改变。



图像格式。

图像大小(1K=1025字节)。

Photoshop Elements甚至会告诉你从拨号modem到浏览器中传送所花的时间。

熟悉质量设定和各种图像格式的最佳途径是实践。你会找到图像和你设计的网页类型所需的质量等级。你同样会学到怎么使用不同于JPEG的其他格式。

问： JPEG选项对话框中质量标签旁边的数字30是干什么用的？

答： 30是Photoshop Element认为质量为“中”。JPEG通常使用1%～100%的比例，对应低、中、高及其他。这是许多图像编辑软件都事先调整好的值。

问： 我可以用<img>元素中的width和height属性来调整图像大小吗？

答： 可以，但这不是一个好主意。为什么？因为如果你这么做，你需要下载整个图像，让浏览器调整图像大小(就像你的浏览器拥有自动调整功能并支持这么做)。width和height属性帮助浏览器指出需要多大空间来保存图像；如果你使用了，它们会与图像的实际宽度和高度匹配。

修改myPod HTML

一旦你保存好图像，就可以退出Photoshop Elements。现在你所要做是修改myPod “index.html” 页面，得到含有 “seattle_med.jpg” 图像的新版本。以下是 “index.html” 文件的片断，只显示你要修改的部分。

```
<html>
   <head>
      <title>myPod</title>
      <style type="text/css">
         body { background-color: #eaf3da;}
      </style>
   </head>
   <body>
      .
      .
      .
      <h2>Seattle, Washington</h2>
      <p>
         Me and my iPod in Seattle!  You can see rain clouds and the
         Space Needle.  You can't see the 628 coffee shops.
      </p>

      <p>
        <img src="photos/seattle_med.jpg" alt="My iPod in Seattle, WA">
      </p>

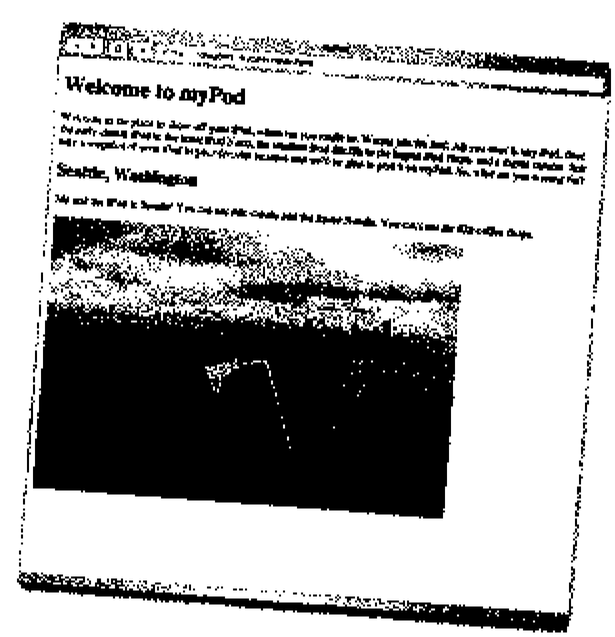
   </body>
</html>
```

HTML的剩余部分在这里。它们已经在你的 “index.html” 文件中。

你所要做的是把<img>元素中的文件名改为你刚刚创建的 “seattle_med.jpg” 。

现在做测试

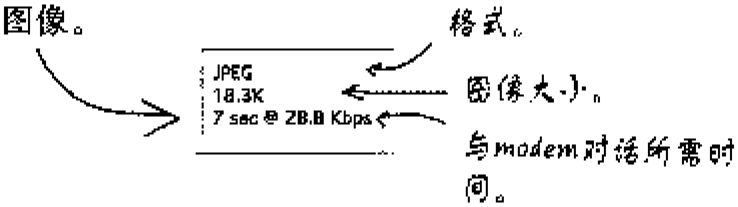
修改并保存，然后在浏览器中重载 “index.html” 。看起来应该好多了。现在图像的尺寸让浏览者舒服多了——没有加载大的无法承受的图像。

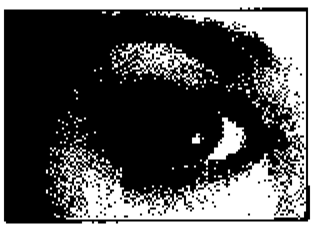
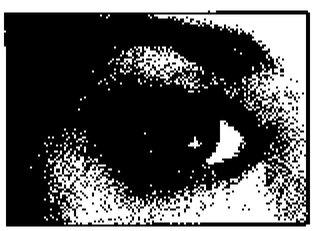
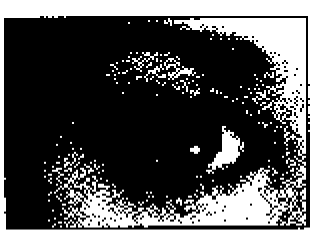
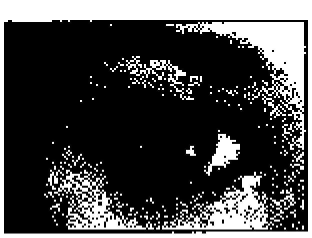


现在图像非常适合浏览器窗口。

用哪种图像格式？

这次你的任务是：用Photoshop Elements打开文件“chapter/testimage/eye.jpg”。打开“保存为Web所用格式”对话框，通过选择JPEG的质量设定，填写以下的空白(低，中，高之类)。你可以在预览面板找到相关信息。完成后，决定哪种设定最适合这种图像。



	Format	Quality	Size	Time	Winner
	JPEG	Maximum			☐
	JPEG	Very High			☐
	JPEG	High			☐
	JPEG	Medium			☐
	JPEG	Low			☐
	GIF	N/A			☐

myPod的更多图像

一批myPod的新图像送到了：两张来自Seattle，一些来自一位在英国的朋友。这些图像都已经调整好了，宽度小于800像素。为它们添加<img>元素(你会在photos文件夹里找到它们)。

```
<html>
  <head>
    <title>myPod</title>
    <style type="text/css">
      body { background-color: #eaf3da;}
    </style>
  </head>
  <body>

    <h1>Welcome to myPod</h1>
    <p>
      Welcome to the place to show off your iPod, wherever you might be.
      Wanna join the fun? All you need is any iPod, from the early classic
      iPod to the latest iPod Nano, the smallest iPod Shuffle to the largest
      iPod Photo, and a digital camera.  Just take a snapshot of your iPod in
      your favorite location and we'll be glad to post it on myPod. So, what
      are you waiting for?
    </p>

    <h2>Seattle, Washington</h2>
    <p>
      Me and my iPod in Seattle!  You can see rain clouds and the
      Space Needle.  You can't see the 628 coffee shops.
    </p>

    <p>
     <img src="photos/seattle_med.jpg" alt="My iPod in Seattle, WA">
     <img src="photos/seattle_shuffle.jpg" alt="An iPod Shuffle in Seattle, WA">
     <img src="photos/seattle_downtown.jpg" alt="An iPod in downtown Seattle, WA">
    </p>

    <h2>Birmingham, England</h2>
    <p>
      Here are some iPod photos around Birmingham. We've obviously got some
      passionate folks over here who love their iPods. Check out the classic
      red British telephone box!
    </p>

    <p>
     <img src="photos/britain.jpg" alt="An iPod in Birmingham at a telephone box">
     <img src="photos/applestore.jpg" alt="An iPod at the Birmingham Apple store">
    </p>
  </body>
</html>
```

也可以在这里轻松地添加些你自己的照片，记住先要调整它们的尺寸。

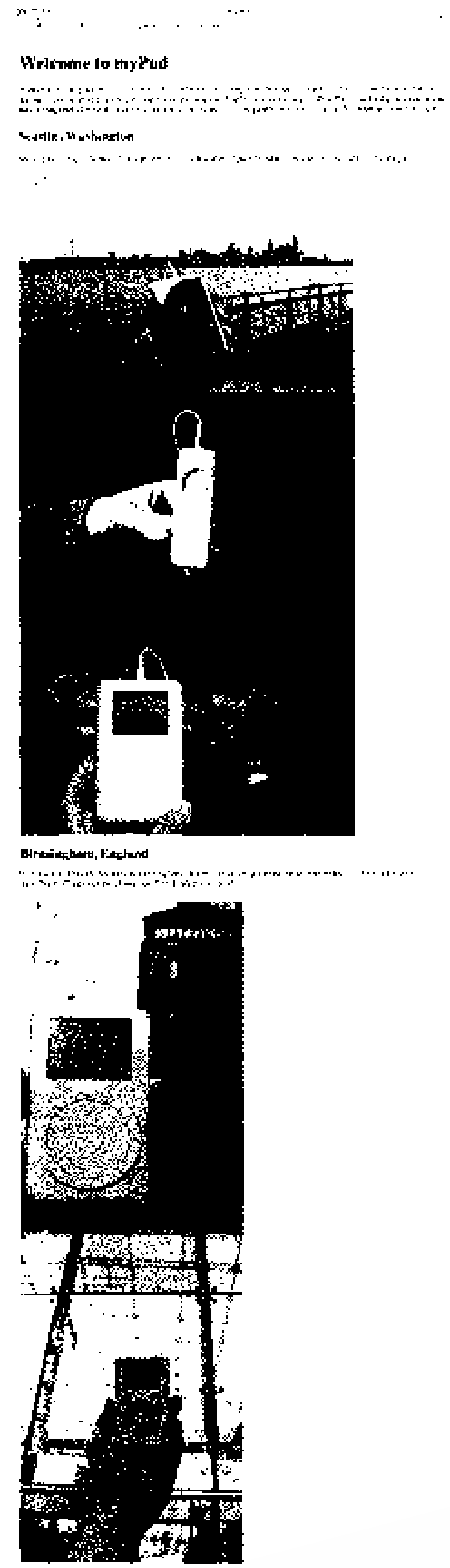
把Seattle（西雅图）的照片放在一起。

同样让Birmingham(伯明翰)的在一起……

对myPod的另外一个测试

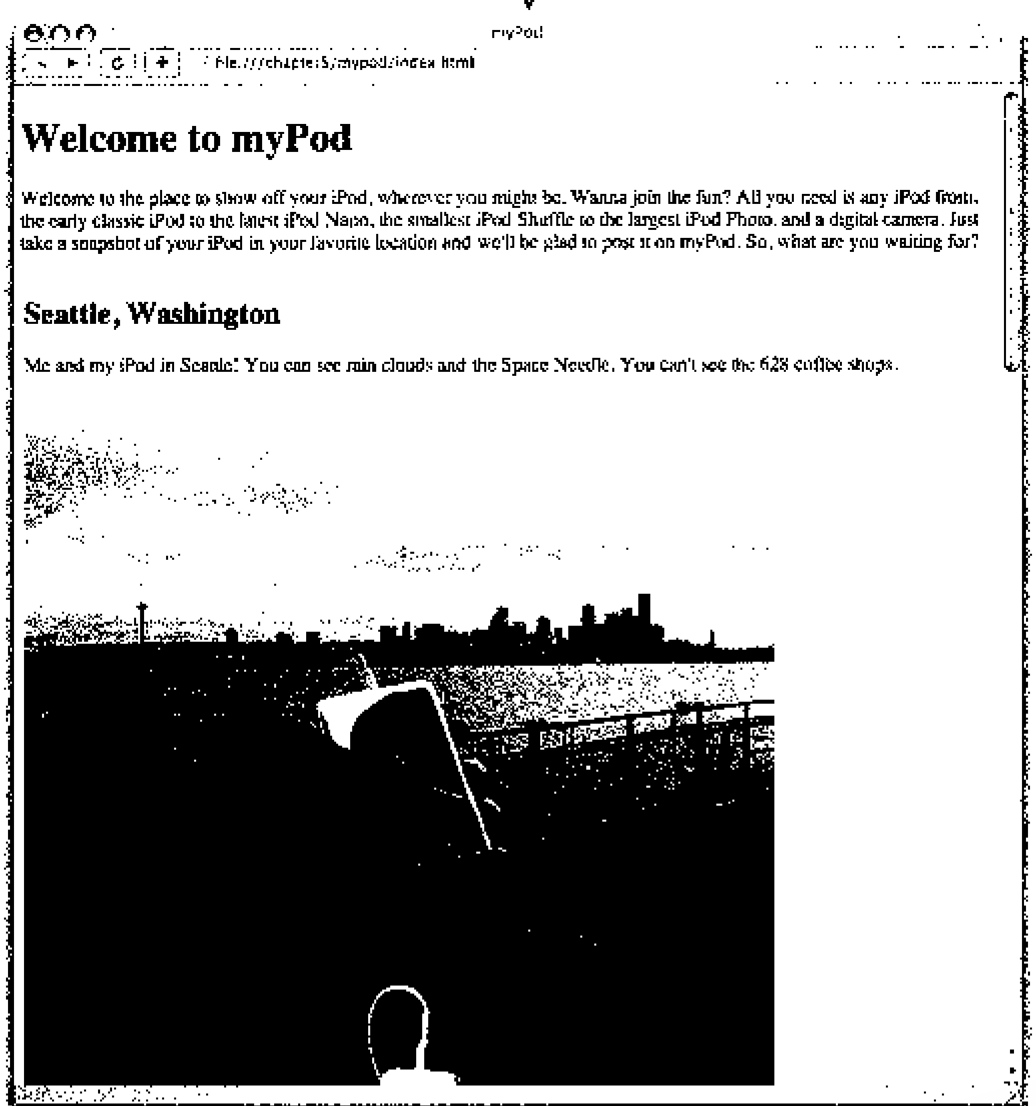
这时我们不需要告诉你用浏览器重载页面了：你肯定已经在我们说之前就这样做了。喔，这些图像令网页如此与众不同，你不这么认为吗？这个页面开始显得有趣了。

但这并不意味着你完成了。尽管网页上已经有了不少图像，还调整了它们的大小，但图像还是太大了。而且当你添加更多图像后，不但网页加载会越来越慢，用户还得滚动翻页来浏览全部的图像。如果用户可以看到一个每张图像的缩略图，然后点击缩略图来看到大图像，不是更好吗？



这里是整个页面和所有图像的样子。

这里是页面当前的样子，关闭它。



使用缩略图重新修改网站

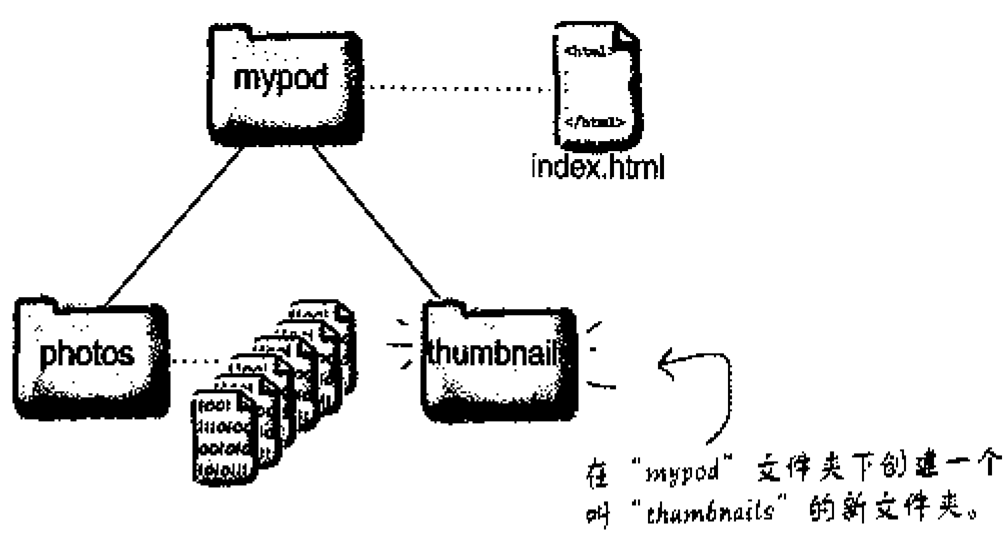
现在你要把每张图像换成一个小图像使网页更具可用性(就是我们所说的缩略图)，然后你要创建缩略图到每张大图像的链接。这里是你要做的，一步一步来：

1. 为缩略图创建一个新文件夹。
2. 调整每张图像为150像素×100像素并保存到“thumbnail”文件夹中。
3. 把“index.html”中的每个`<img>`元素的`src`设置成图像的缩略版本。
4. 添加从每个缩略图到包含大图像的新页面的链接。

为缩略图创建一个新文件夹

为了便于组织，为缩略图创建一个单独的文件夹。否则，你会看到一个文件夹内的大图像和小缩略图混合在一起，这样在你加入大量图像后会非常混乱。

在mypod文件夹下创建一个叫“thumbnails”的新文件夹。如果你用的是书上的示例文件，你会发现文件夹已经存在了。



创建缩略图

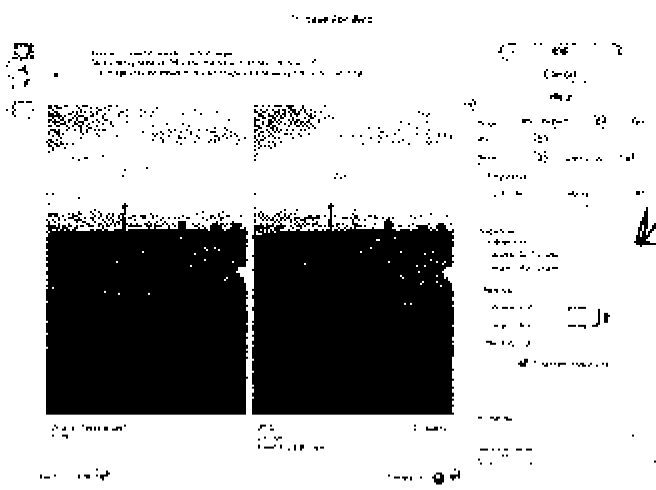
你已经有地方放置缩略图了，我们开始创建吧。先用你的图像编辑软件打开“seattle_med.jpg”文件，你要用和创建600像素×400像素相同的方法来调整成150像素×100像素。

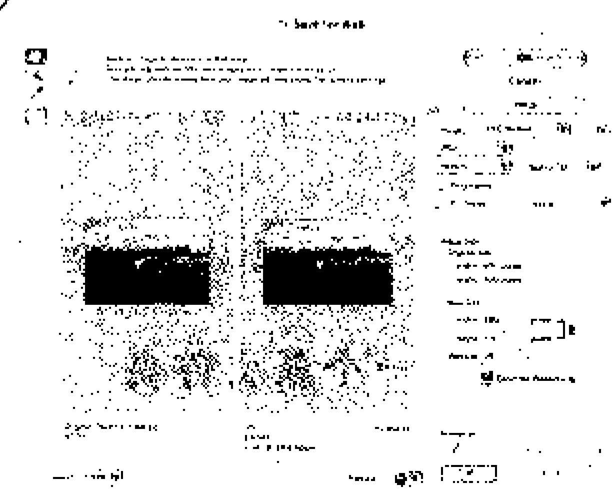
在Photoshop Elements中，选择“保存为Web所用格式”菜单选项。

然后改变宽度为150，高度为100，并单击“应用”按钮。

不要忘记改变格式为JPEG，质量为中等。

最后，单击“OK”按钮。





当图像调整完成后，单击“OK”，并在thumbnails文件夹里以相同名字保存。小心：如果保存在“photos”文件夹下，会覆盖掉大图像。

现在，对你“ photos”文件夹下的每张图像重复这些步骤。

如果你用的是书上的示例文件，你会发现缩略图都存在“thumbnails”文件夹里。无需对所有图像进行处理(毕竟，你学习的是HTML，不是图像的批处理)。



那Birmingham的图像怎么办——他们的高度比宽度大。150×100有意义吗？

好眼力。因为这些图像的高度比宽度大，我们有两个选择：我们可以改变尺寸使它们变成100×150或者可以剪裁每张图像从中获得150×100的缩略图。我们要制作自己的150×100图像；如果你想弄清你的图像编辑软件是如何完成的，你可以随意剪裁并创建150像素×100像素图像。



用缩略图重新设计HTML

现在你需要修改HTML，使**<img>**元素从“thumbnails”文件夹而不是“photos”文件夹中获得图像。并且，因为你现在使用相对路径，像“photos/seattle_med.jpg”，那会比较简单：你要做的是把每个**<img>**元素的“photos”文件夹名改成“thumbnails”。

```
<html>
  <head>
    <title>myPod</title>
    <style type="text/css">
      body { background-color: #eaf3da;}
    </style>
  </head>
  <body>

    <h1>Welcome to myPod</h1>
    <p>
      Welcome to the place to show off your iPod, wherever you might be.
      Wanna join the fun? All you need is any iPod, from the early classic
      iPod to the latest iPod Nano, the smallest iPod Shuffle to the largest
      iPod Photo, and a digital camera.  Just take a snapshot of your iPod in
      your favorite location and we'll be glad to post it on myPod. So, what
      are you waiting for?
    </p>

    <h2>Seattle, Washington</h2>
    <p>
      Me and my iPod in Seattle!  You can see rain clouds and the
      Space Needle.  You can't see the 628 coffee shops.
    </p>
```

你所要做的是把文件夹名由“photos”改为“thumbnails”。

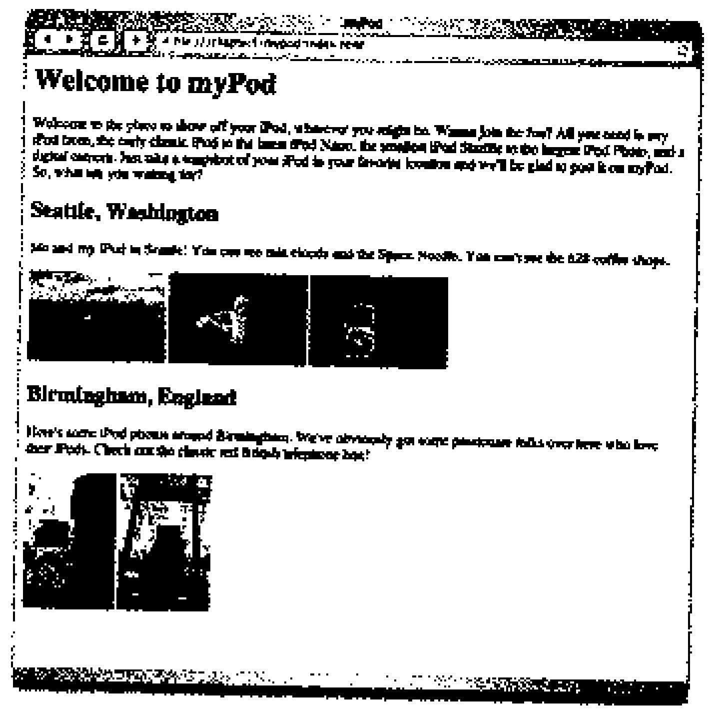
```
    <p>
      <img src="thumbnails/seattle_med.jpg" alt="My iPod in Seattle, WA">
      <img src="thumbnails/seattle_shuffle.jpg" alt="A iPod Shuffle in Seattle, WA">
      <img src="thumbnails/seattle_downtown.jpg" alt="An iPod in downtown Seattle, WA">
    </p>

    <h2>Birmingham, England</h2>
    <p>
      Here are some iPod photos around Birmingham. We've obviously got some
      passionate folks over here who love their iPods. Check out the classic
      red British telephone box!
    </p>

    <p>
      <img src="thumbnails/britain.jpg" alt="An iPod in Birmingham at a telephone box">
      <img src="thumbnails/applestore.jpg" alt="An iPod at the Birmingham Apple store">
    </p>
  </body>
</html>
```

对myPod的另一个测试

啊哈……网站又改进好多。浏览者一眼就可以看到所有图像了。他们可以更轻松地分清哪张图像是哪个城市的。现在我们要指定一个从缩略图链接到相应的大图像的路径。





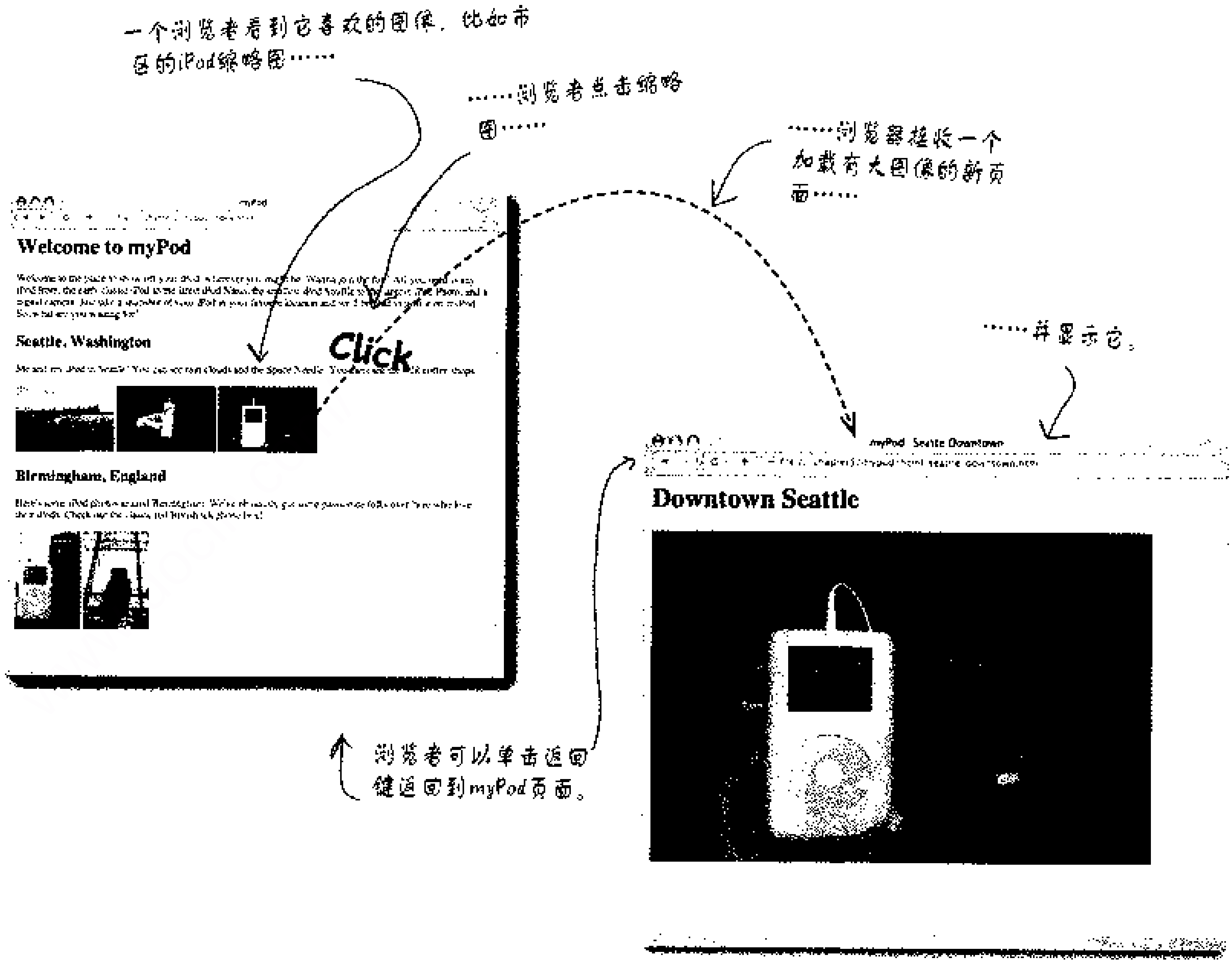
对；但是记住<img>元素是个内联元素。

换句话说，我们没有“落下任何东西”。因为**<img>**元素是个行元素，显示时前边或相邻边不会有换行。所以，假如你的HTML有几张图像，如果浏览器窗口足够宽，他们会并排显示。

大图像不会并排显示的原因是浏览器没有足够空间，所以会纵向显示。浏览器通常在块元素垂直方向上显示空白，如果你回头查看屏幕截图，你会看到图像纵向之间没有任何空白。这是**<img>**作为内联元素的又一个标识。

让缩略图变成链接

你快要完成了。现在你只需要创建从每个缩略图到大版本图像的链接。下面是做法示范：



你需要以下内容来完成工作：

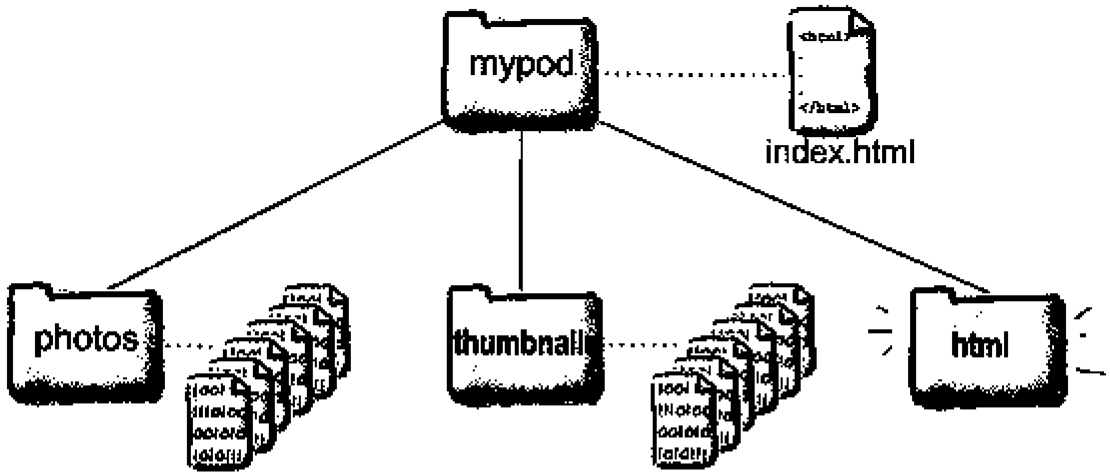
❶ **一个显示每张带标题的图像的页面，标题描述了图像的内容。**

❷ **从每张在“index.html”中的缩略图到对应的大图像的链接。**

我们先创建网页，再回来完成链接。

为图像创建单独的页面

首先，在“mypod”文件夹下，创建一个名为“html”的新文件夹来保存那些单独的页面。



现在我们要为每张图像创建一个HTML文件。如果图像叫做“seattle_med.jpg”，我们给HTML文件起名为“seattle_med.html”以保持一致。每个HTML文件中，在图像后边都有一个描述图像的标题。下面是第一个Seattle图像的HTML。其他的页面全部都采用这种格式。

页面的标题，用来描述照片。

这是我们已经做好的CSS，这样可以使页面颜色一致。

```
<html>
  <head>
    <title>myPod: Seattle Ferry</title>
    <style type="text/css"> body { background-color: #eaf3da; } </style>
  </head>
  <body>
    <h1>Seattle Ferry</h1>
    <p>
      <img src="../photos/seattle_med.jpg" alt="An iPod on the Seattle ferry">
    </p>
  </body>
</html>
```

这里我们给页面一个描述性的标题。

这是指向大的“seattle_med.jpg”图像的<img>元素。我们用alt属性描述图像。

注意我们需要在相对路径中使用“..”，因为“html”文件夹是“photos”文件夹的兄弟，所以当要使用相对链接时，我们需要上溯一个文件夹再下探到“photos”。

练习

看看本章示例文件中的“html”文件夹，你会发现所有单独图像页面已经存在，除了“seattle_downtown.jpg”页面。在“html”文件夹下创建一个名为“seattle_downtown.html”的页面，并测试它。在你继续操作前先确保它能正常工作。如果有任何问题，你会在这章后边找到答案。

怎么创建图像的链接呢？用images?

你已经有大图像、小缩略图和一批显示单独图像的HTML页面。现在你需要把他们放到一起并把在“index.html”中的缩略图链接到“html”文件夹下的页面。怎么做呢？

要链接图像，你要在**<a>**元素中放置**<img>**元素，如下所示：

这是在<img>元素之前的一个<a>开始标记。

Href为照片链接到新的HTML页面——“seattle_downtown.html”在“html”目录中。

这里是供“seattle_downtown”缩略图用的<img>元素，就像在“index.html”里一样。

```
<a href="html/seattle_downtown.html">
    <img src="thumbnails/seattle_downtown.jpg"
         alt="An iPod in downtown Seattle, WA">
</a>
```

这是<a>结束标记。

<img>元素完全嵌套在<a>元素中。

你把**<img>**元素放置到**<a>**中后，浏览器会把图像当作可点击的链接。当你点击图像，浏览器会接收**href**中的页面。

添加图像链接到“index.html”

这是最后一步。你需要在“index.html”文件中将<a>元素添加在缩略图的<img>元素周围。记住，每个<a>元素中的href应该链接到“html”文件夹中包含大版本图像的页面。确保你的链接、缩略图和页面都正确匹配。

这是完成的“index.html”文件。你需要做的是添加灰色的HTML。

```
<html>
   <head>
      <title>myPod</title>
      <style type="text/css">
         body { background-color: #eaf3da;}
      </style>
   </head>
   <body>

      <h1>Welcome to myPod</h1>
      <p>
         Welcome to the place to show off your iPod, wherever you might be.
         Wanna join the fun? All you need is any iPod, from the early classic
         iPod to the latest iPod Nano, the smallest iPod Shuffle to the largest
         iPod Photo, and a digital camera.  Just take a snapshot of your iPod in
         your favorite location and we'll be glad to post it on myPod. So, what
         are you waiting for?
      </p>

      <h2>Seattle, Washington</h2>
      <p>
         Me and my iPod in Seattle!  You can see rain clouds and the
         Space Needle.  You can't see the 628 coffee shops.
      </p>

      <p>
      <a href="html/seattle_med.html">
         <img src="thumbnails/seattle_med.jpg" alt="My iPod in Seattle, WA">
      </a>
      <a href="html/seattle_shuffle.html">
         <img src="thumbnails/seattle_shuffle.jpg" alt="A iPod Shuffle in Seattle, WA">
      </a>
      <a href="html/seattle_downtown.html">
         <img src="thumbnails/seattle_downtown.jpg" alt="An iPod in downtown Seattle, WA">
      </a>
      </p>

      <h2>Birmingham, England</h2>
      <p>
         Here are some iPod photos around Birmingham. We've obviously got some
```

```
        passionate folks over here who love their iPods. Check out the classic
        red British telephone box!
      </p>

      <p>
      <a href="html/britain.html">
        <img src="thumbnails/britain.jpg" alt="An iPod in Birmingham at a telephone box">
      </a>
      <a href="html/applestore.html">
        <img src="thumbnails/applestore.jpg" alt="An iPod at the Birmingham Apple store">
      </a>
      </p>
    </body>
</html>
```

用<a>元素包围每个缩略图。注意在每个链接中添加正确的href。

添加**<a>**元素到你的“index.html”文件。保存，用浏览器加载并检验myPod!

there are no Dumb Questions

问： **我用<a>元素包围一段文字时会有下划线。为什么图像不一样呢？**

答： 确切来说，大多数浏览器会在可链接的图像周围添加边框。(我们使用的浏览器Safari，是其中一个不添加的。)如果浏览器在你的链接图像上添加了边框，而你又不喜欢，在后边的章节你会学到如何用CSS去掉边框。同样你会注意到当把鼠标悬停在图像上时，光标会提示你链接图像的简要信息。大多数情况下即使图像没有边框，你的用户也可以通过鼠标或上下文来获知它是可链接的。

问： **我们必须用那些HTML页面而不能直接链接到JPEG图像吗？我想浏览器完全有能力自己来显示图像。**

答： 你说对了，你可以直接链接到图像，像这样：……。如果你这么做并点击链接，浏览器会在空白页上自己显示图像。通常认为直接链接到图像不太好，因为你通常想为显示的图像提供一些上下文。



好主意。实际上，我们已经做好了一个myPod logo。

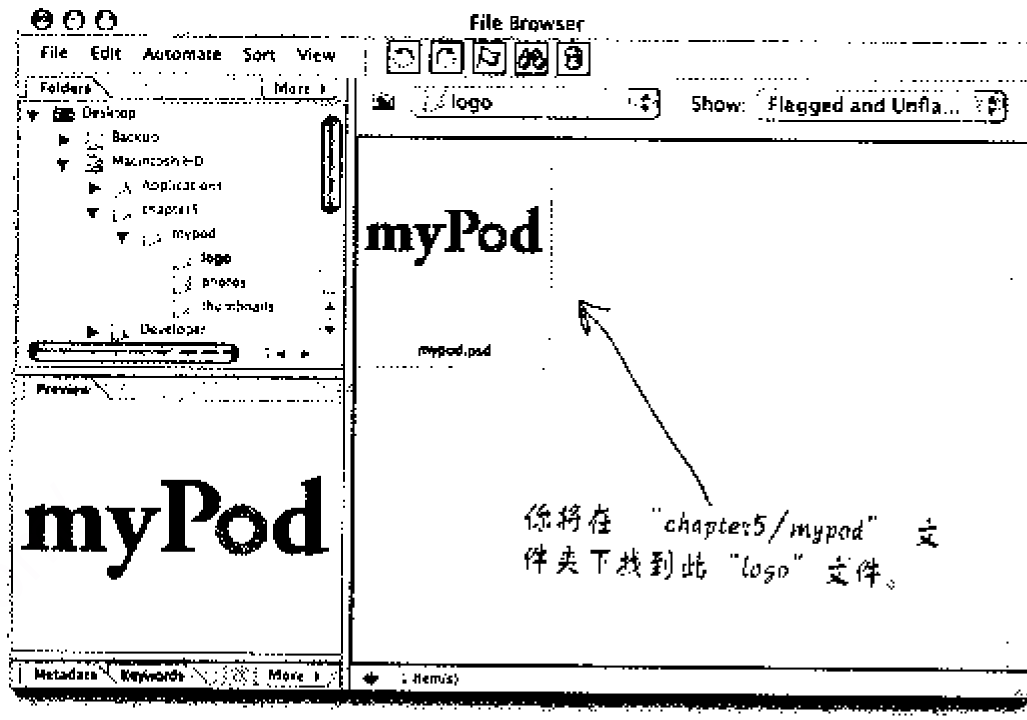
再看看“chapter5/mypod”文件夹，你会找到一个名为“logo”的文件夹。在这个文件夹里你可以找到“mypod.psd”文件。“psd”意思是文件以Photoshop 格式保存，这是Photoshop对数字图像的常用格式。但是Photoshop格式文件是为处理数字图像而设置的，不是为网页，所以我们要做些工作来从中获得“为网站而设”的图像。

许多图像编辑软件支持.psd文件，所以就算你没有Photoshop Elements，也可以继续后面几页的操作步骤。如果你的软件不能打开.psd文件，你会在“logo”文件夹下找到每步的图像。

打开myPod logo

让我们看看myPod logo：用Photoshop Elements打开“chapter5/mypod/logo”文件夹中的“mypod.psd”文件。

如果你的图像编辑软件不能打开文件，跟我们继续以下的步骤——相同的原理对不同格式一样奏效。

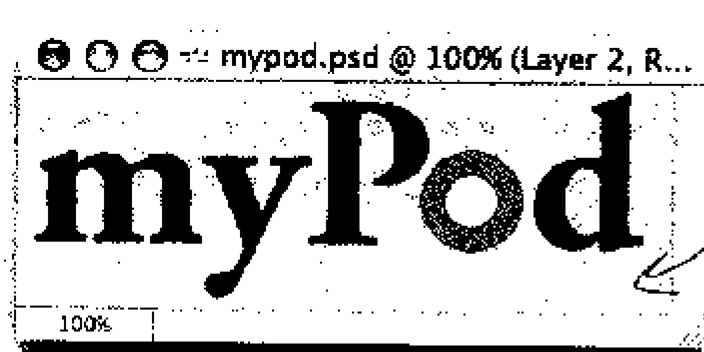


你将在“chapter5/mypod”文件夹下找到此“logo”文件。

仔细观察……

漂亮的logo：两个圆结合了几种印刷格式，一个灰色一个白色(显然灵感来源于iPod的操作导向轮)。

但是背景中的选定样式是什么？这是大多数图像编辑软件显示透明区域的方式。记住它，我们会给logo选择一个图像格式……

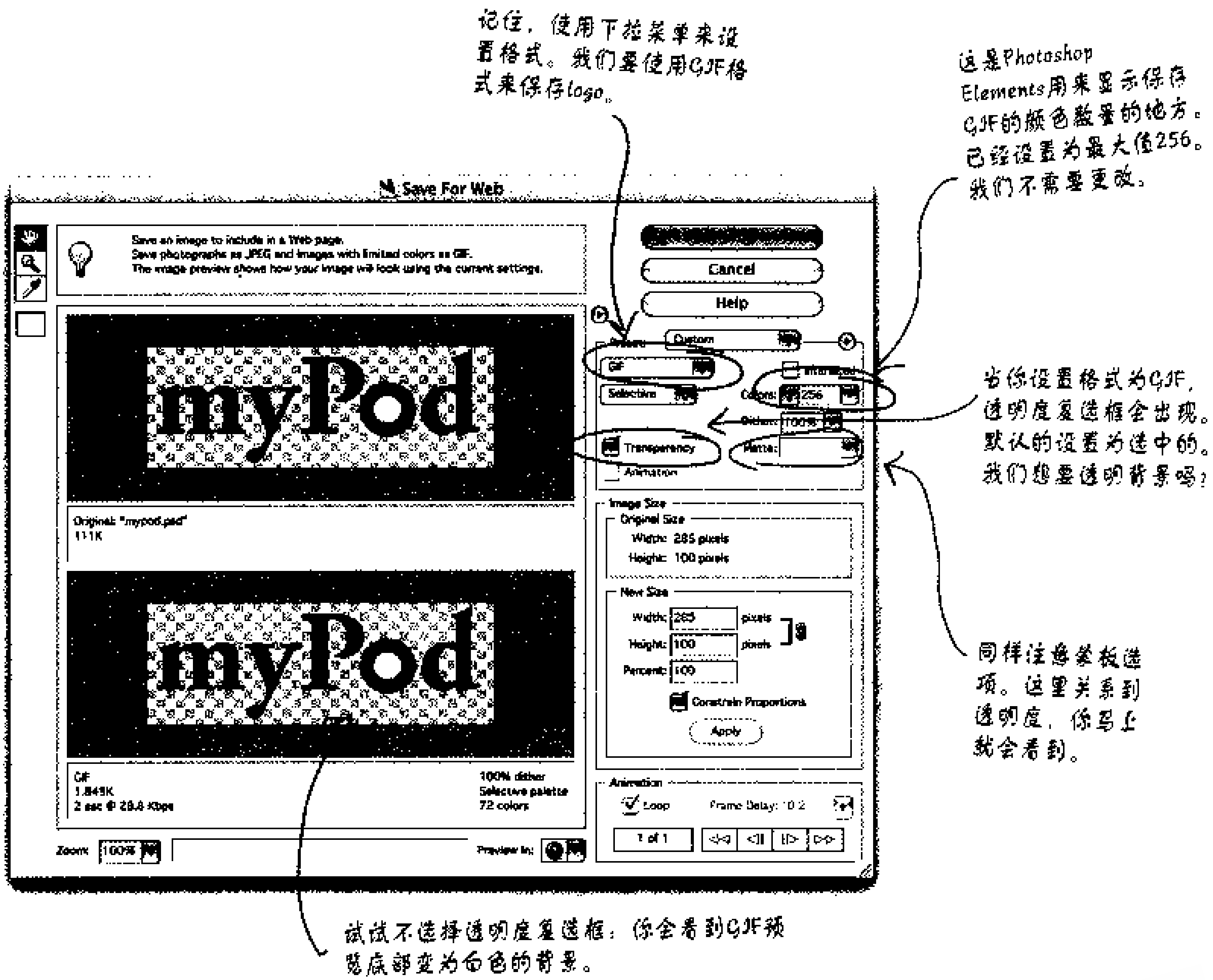


无论何时你看这个选定样式，图像中都会显示一个透明区域。

我们应该用哪种格式？

你已经知道我们有许多格式用来保存图像：我们可以使用JPEG或者GIF。这个logo仅仅使用三种颜色，还有文字和一些几何图形。从你所学的两种图像格式中，你比较看好GIF（好选择！）。

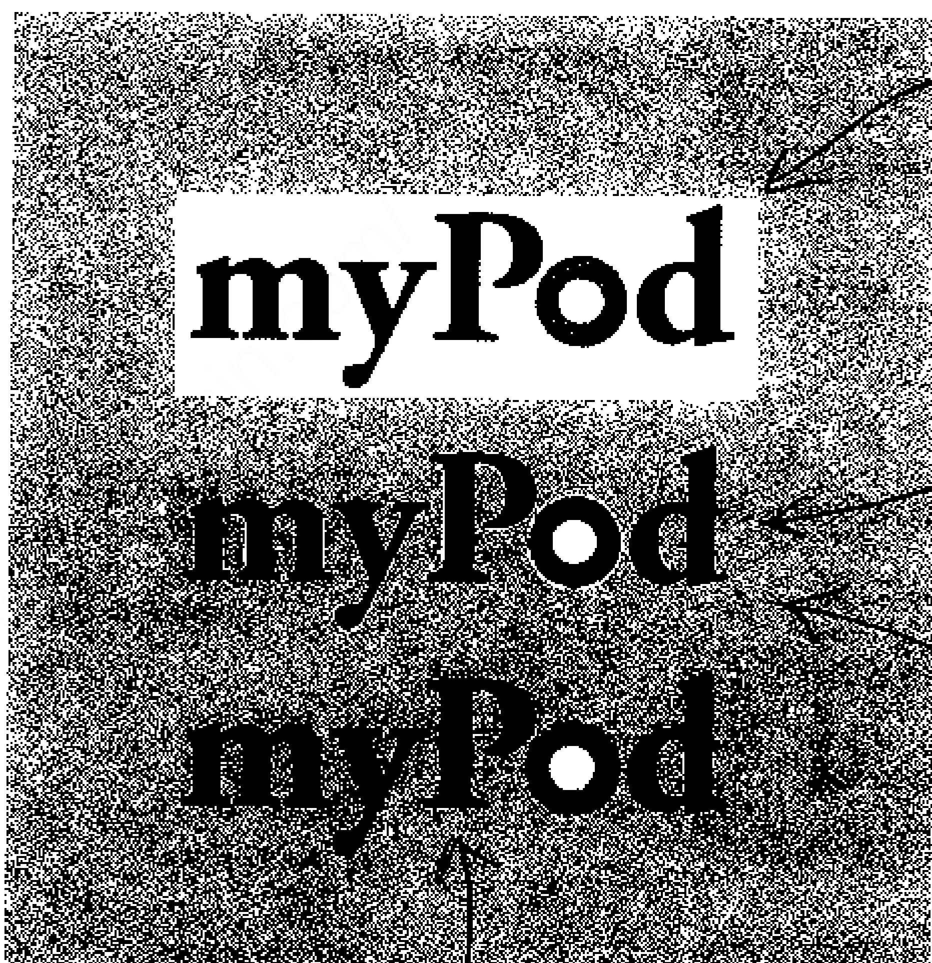
那继续并选择GIF格式，你会看到我们有更多选择。我们一起来看看吧……



透明还是不透明？这是个问题……

MyPod logo放置在淡绿色的背景上，你可能觉得还是透明比较好看，对不对？选择“保存为Web所用格式”对话框里的一些选项，并比较使用后的logo。

这是logo用三种不同的方式保存后在网页的绿色背景上显示的结果。



不用透明度看起来真差劲。很明显，白色背景不适合绿色的网页（然而，也许它在白色的网页上会很好）。

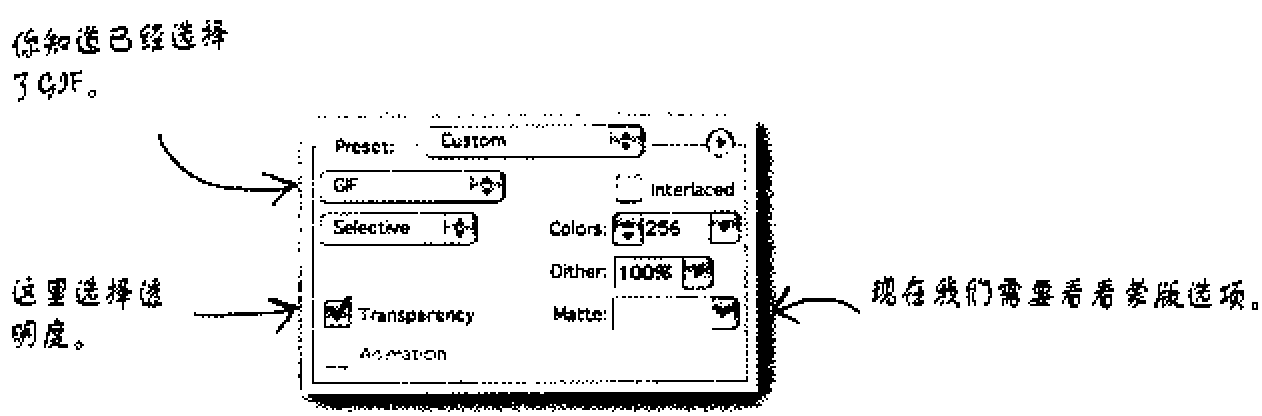
这里我们选择了透明度并保存。好一点儿……但是logo里字母周围的白色“光轮”是什么？

光轮是因为图像编辑软件创建了蒙板来柔化文字边缘和背景颜色。这么做以后，它认为它柔化了<u>白色</u>背景下的边缘。

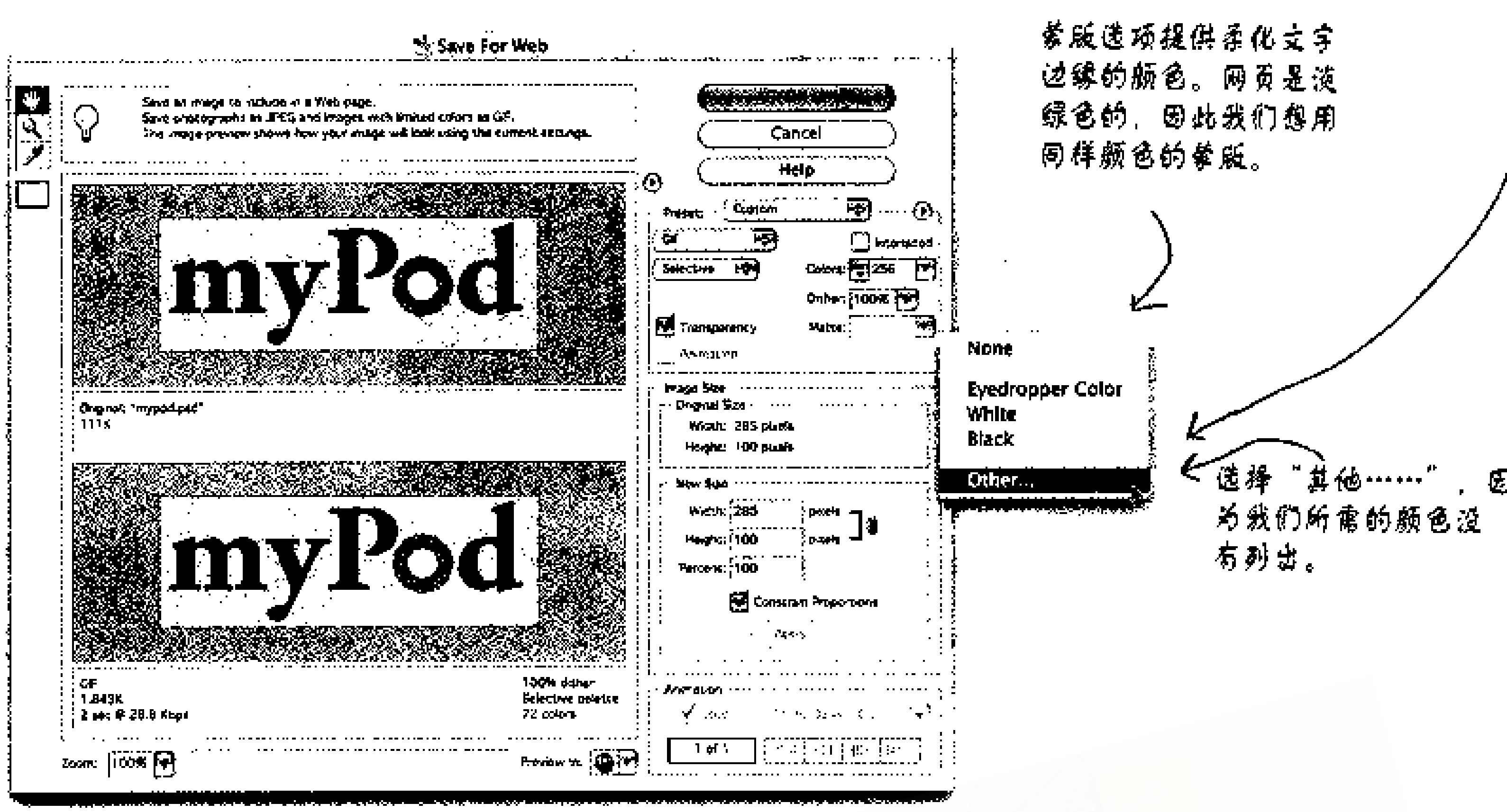
啊，现在好了，看起来非常棒。对于这个版本我们告诉Photos Elements在文字周围用<u>绿色</u>背景创建了蒙版。怎么做的？马上告诉你。

保存透明的GIF

你知道我们想要logo的透明GIF版本，你还知道我们需要使用蒙版来去掉文字周围的光轮。让我们看看GIF的“保存为Web所用格式”的对话框。



蒙版选项允许你选择文字周围蒙版的颜色。我们想让它成为网页的背景颜色。



等等，什么是网页的背景颜色？

还记得在myPod的“index.html”文件中我们做好的CSS吗？那个CSS是设置网页的背景颜色为淡绿的。我们这样获得颜色：

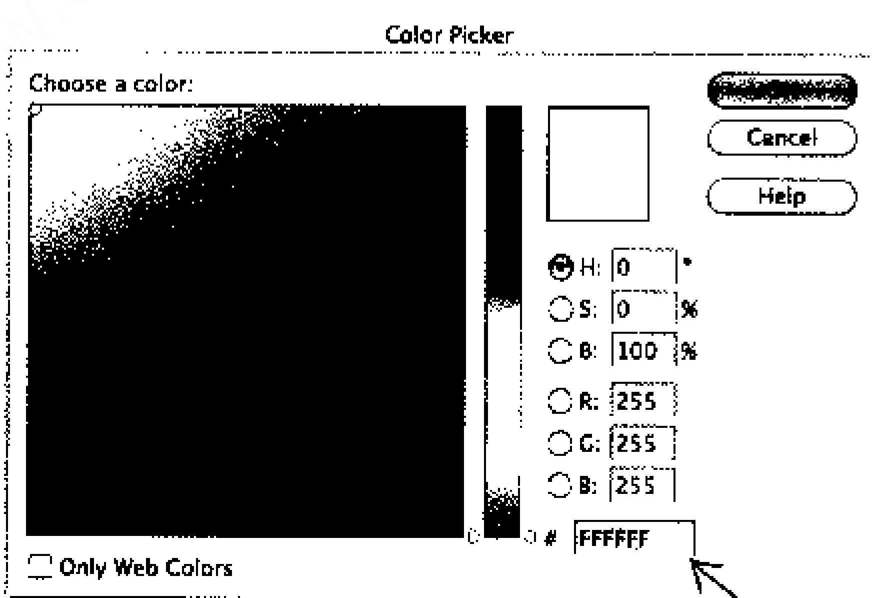
```
<style type="text/css">
   body { background-color: #eaf3da;}
</style>
```

这里是背景颜色。

什么？你不知道这是淡绿色？现在按我们所说的做，我们会在后面的章节中再回头解释关于颜色的一切。

设置蒙版颜色

当你点击了蒙版下拉菜单并选择“其他……”菜单选项，Photoshop Elements会打开调色板对话框。

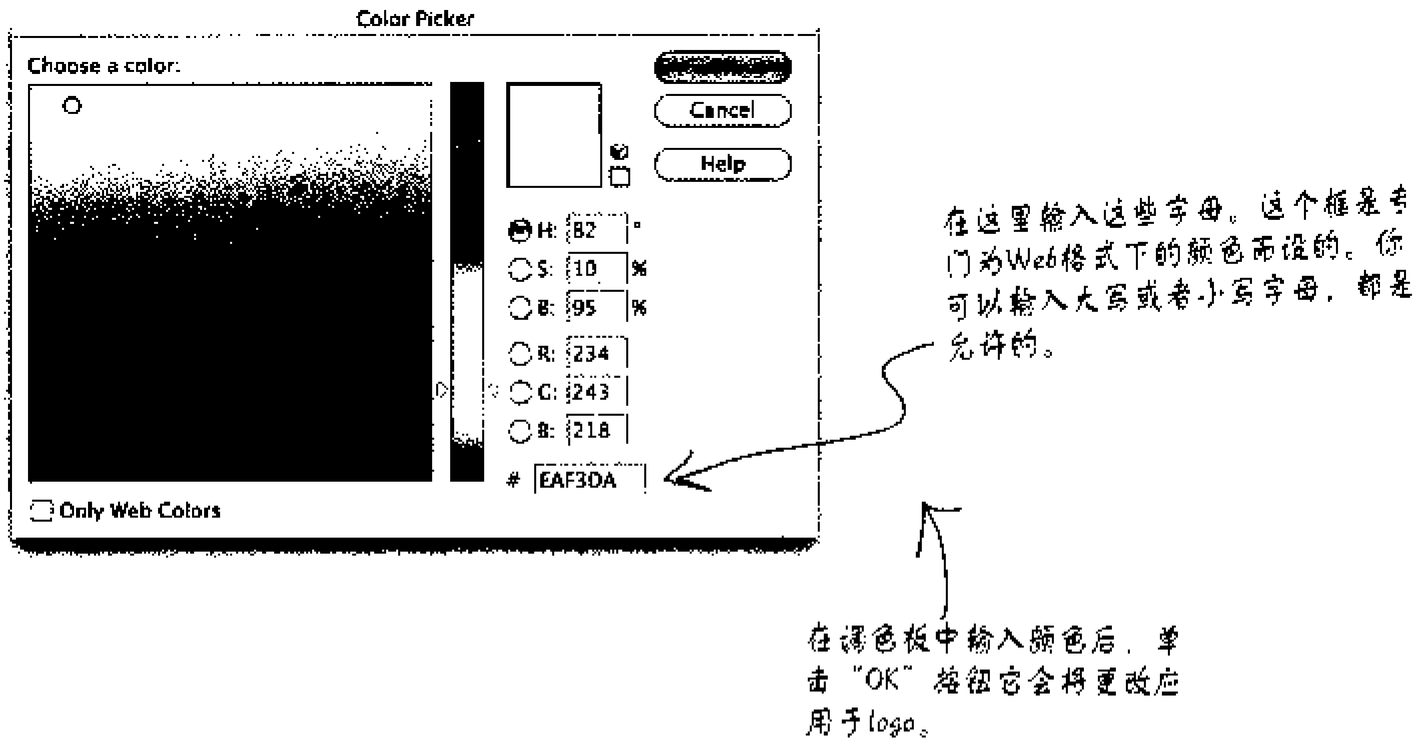


调色板给你许多不同方式来选择蒙版颜色。我们只是想设置网页的背景颜色，我们已经知道它是eaf3da……

这里就是我们需要的颜色。

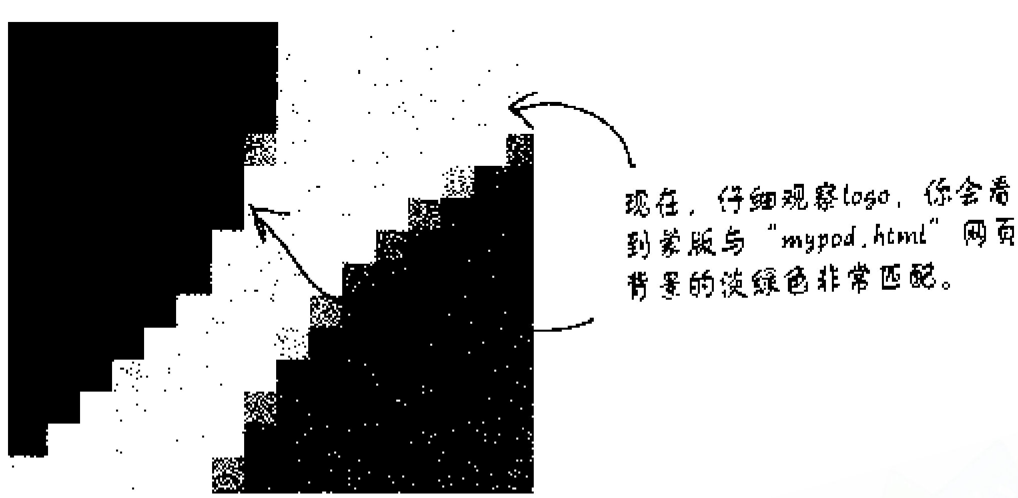
继续设置蒙版颜色

继续并在调色板对话框中输入颜色“eaf3da”。你会看到颜色改变为myPod页面的背景颜色。



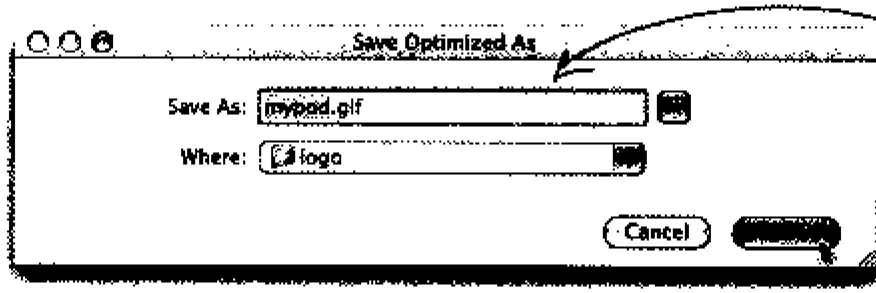
用蒙版做logo

现在在预览面板中仔细观察一下logo。你会看到Photoshop Elements已经在生硬的边缘添加了淡绿色蒙版，在网页上显示时，让myPod拥有一个更加柔和、光滑的外观。



保存logo

OK，你已经完成了在“保存为Web所用格式”对话框中所需的调整，现在继续单击“OK”按钮，保存图像名为“mypod.gif”。



Elements会自动把文件名后缀改为“.gif”。在“logo”文件夹下保存图像为“mypod.gif”。

添加logo到myPod网页

现在你所要做的是添加logo到myPod网页。我们会把它添加到顶部的网站简介和iPod图像上。这样，浏览者浏览myPod网页时第一眼就能看到它。

```
<html>
   <head>
      <title>myPod</title>
      <style type="text/css">
         body { background-color: #eaf3da;}
      </style>
   </head>
   <body>
      <p>
         <img src="logo/mypod.gif" alt="myPod Logo">
      </p>

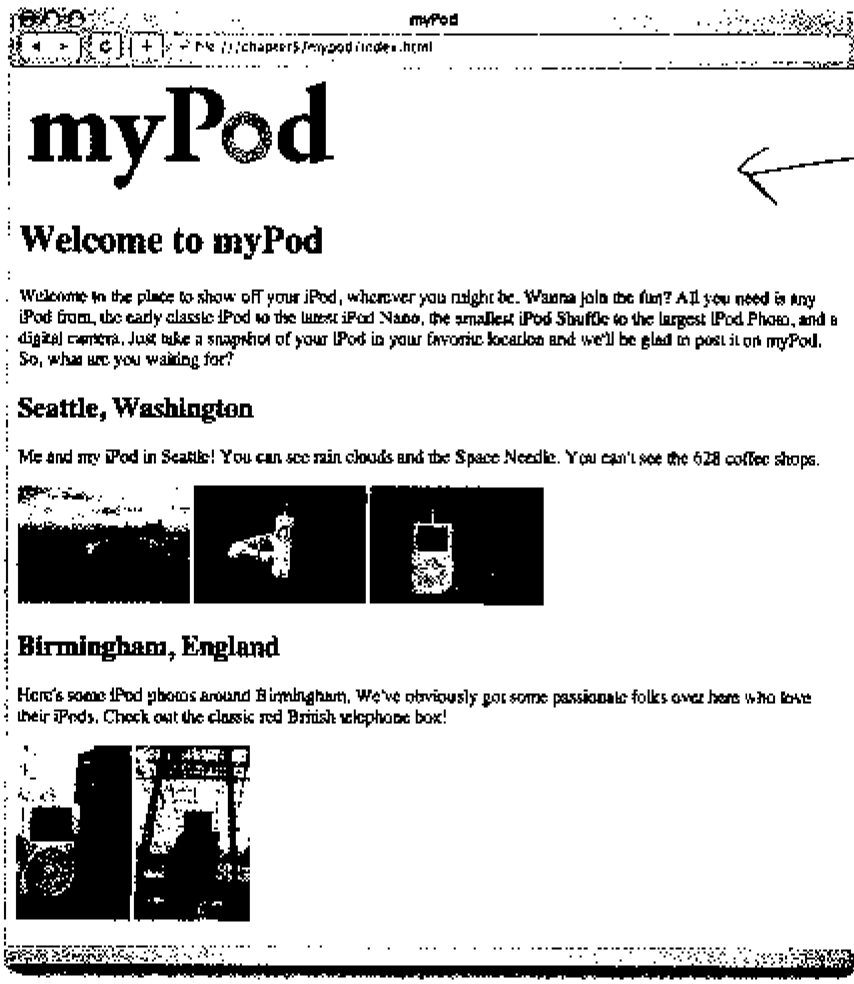
      <h1>Welcome to myPod</h1>
      .
      .
      .
   </body>
</html>
```

添加logo图像到myPod网页的顶部。记住为logo使用正确相对路径，它在“logo”文件夹中，并添加一个alt属性来介绍图像。

“index.html”中其余的HTML在这里。

现在做最后的测试

我们来测试一下这个小宝贝！用浏览器重载网页并查看你的myPod透明的GIF logo工作的怎么样了。



它正常显示了——一切努力得到了回报。你的myPod网页现在有了一个非常漂亮的logo。



there are no Dumb Questions

问： **要编写精美的网页，我需要知道关于图像格式的一切吗？**

答： 不。不用任何图像你就可以创建精美的网页。然而，图像是网站的一个重要部分，所以一些关于图像如何工作的知识是非常有用的。有时一张或者两张图像就是一个好网站和精美网站之间的差别。这里有许多关于图像的知识，都比较容易学会。

问： **为什么需要柔化文字边缘？**

答： 看看以下两种不同版本的myPod logo：



你会看到上边的版本非常生硬，边界带锯齿而且可读性差。这是文本在电脑屏幕上的默认显示。第二种版本使用一种叫反锯齿处理的技术来柔化边界。反锯齿处理过的文字在电脑屏幕上的显示具有更好的可读性，眼睛看起来也比较舒服。

问： **那“蒙版”有什么用呢？**

答： 用反锯齿处理柔化边缘的过程和背景颜色有关。如果你把下边的logo版本(在前一个问答中)放到一个有颜色的背景中，你会看到它有白色的边缘。在Photoshop Elements中的蒙版选项允许你指定文字所处的背景颜色，所以当文字柔化后可以和背景颜色匹配。

问： **这种技术只能在文字上使用吗？**

答： 不，它能应用在图像中任何会导致锯齿的线条。看看myPod logo的圆圈，它也用了蒙板。

问： **为什么我不能将背景颜色设为纯色来配合网页的颜色？**

答： 你可以这么做，但是有一个缺点：如果你网页上还有其他能透过透明层显示的东西，它们在纯色版本下无法显示。你还没看到这类的例子，但是当我们介绍CSS时，就会看到了。

问： **如果我在使用蒙版后改变背景颜色会怎样呢？**

答： 背景颜色的微小变化可能不会引人注意；然而，如果突兀地改变颜色，你必须用新蒙版颜色来重新创建GIF。

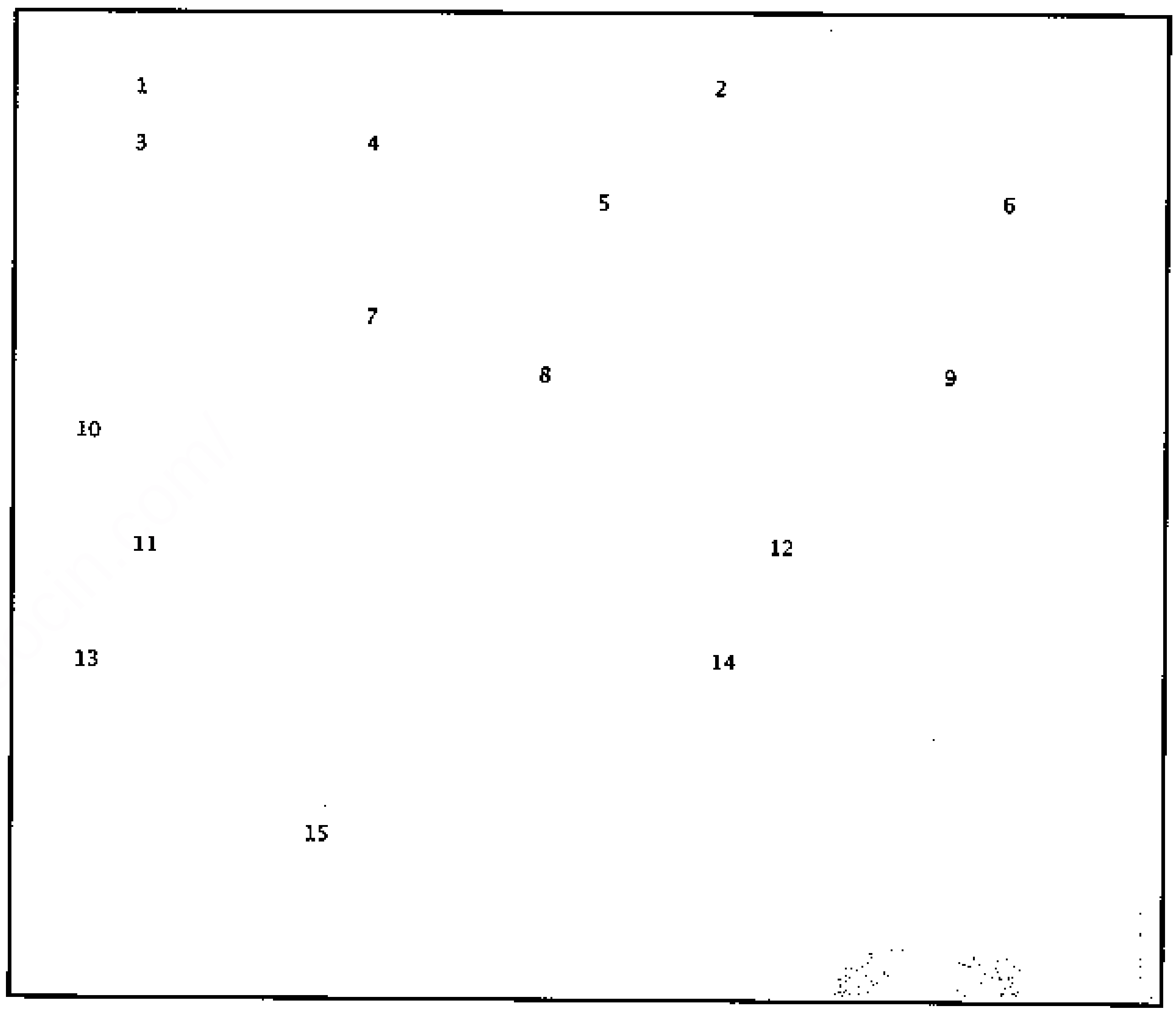
要点

- 使用<img>元素在网页中添加图像。
- 浏览器对待<img>元素不同于其他HTML元素；在读完HTML页面后，浏览器从网络服务器上接收每张图像并显示。
- 如果你网页上有许多大图像，你要创建缩略图来使你的网页更具实用性并更快地下载。用户通过点击小图像来看大版本的图像。
- <img>元素是个内联元素，意味着浏览器不会在图像前边或者后边插入换行。
- src属性用来指定图像文件的位置。你可以在src属性中使用相对路径从你的网站上得到图像，或者用URL从别的网站获得图像。
- <img>元素中的alt属性是对图像意义的描述。如果图像找不到，它会在某些浏览器中显示，并被屏幕读取器用于为视觉有障碍的用户描述图像。
- 宽度小于800像素是网页图像显示的优秀方案。大多数照片是用数码相机拍摄的，对网页来说它们太大，你必须调整它们的大小。
- Photoshop Elements是众多图像编辑软件中可以用来调整图像大小的一种。
- 对浏览器来说图像太大会让网页难以使用、下载和显示变慢。
- 一个像素是屏幕能显示的最小点。每个图像都是由数以千计的像素组成的。视你的显视器而定，可以从72像素/英寸~120像素/英寸不等。
- JPEG和GIF是网络浏览器广泛接受的两种图像格式。
- JPEG格式最适用于照片和其他复杂图像。
- GIF格式最适用于logo和其他由纯色、线条或者文字组成的简单图像。
- JPEG图像可以用多种质量等级压缩，你可以根据需要选择质量和文件大小的平衡。
- GIF图像格式允许你创建一个带透明背景的图像。如果你把一个带透明背景的图像放到网页上，在图像背后的部分如页面的背景颜色，会穿透图像的透明部分显示。
- 在Photoshop Elements中，在“保存为Web所用格式”对话框中使用蒙版颜色菜单，选择正确颜色来柔化透明GIF图像的边缘。
- 图像可以像链接一样链接到其他网页。使用<img>元素作为<a>元素的内容，把链接放到<a>元素中的href属性来创建图像链接。



HTML填字游戏

该到你大脑的右半球放松一下并让左半球开始工作的时间了。这些字谜都与本章的HTML有关。



横排提示：

3. 屏幕上最小的元素。
5. 网络服务器对它们其中的每一个都发送请求。
7. 对单色、线条和小文本比较适合。
9. 新的图像格式。
10. 大多数网络服务器用这种方式接收图像。
11. 你可以用铅笔画出的公里数。
13. 页面上的小图像。
14. 你使用Photoshop Elements来对图像做这个。
15. alt属性可以改进它。

竖排提示：

1. 可爱的MP3播放器。
2. 用JPEG你可以控制它。
4. 图像越大，接收它就越_____。
6. GIF有的，但是JPEG没有。
8. 用来柔化文字边缘的技术。
12. 对连续色调图像比较适合。

练习答案

用哪种图像格式？

祝贺你：你已经当选为今天的“盛大图像格式选择者”。给以下图像选择网上的最佳显示格式。

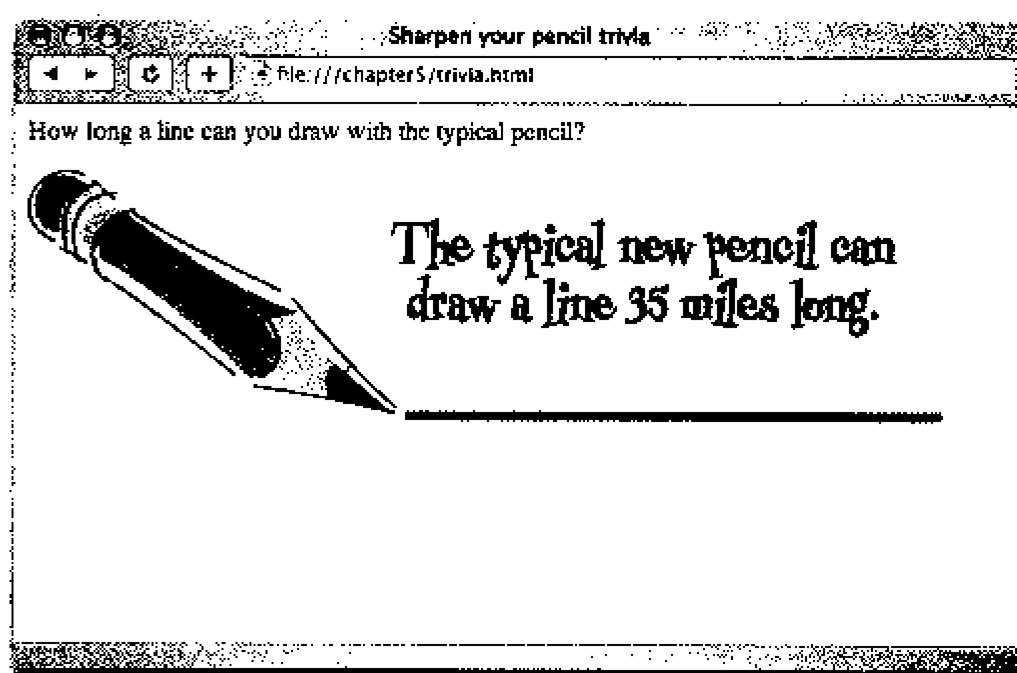
		JPEG	或者 GIF
	带大量连续灰色阴影的图像。	☑	☐
	只有两种颜色和少量文字，当然是GIF。	☐	☑
	带许多颜色的照片，当然是JPEG。	☑	☐
	只有黑和白的图标，GIF。	☐	☑
	这个图像处于边缘。它有一系列连续的颜色(JPEG)，但是同样是几何图形(GIF)，你想用请求透明度的方式来处理它(GIF)。	☑	☑

Sharpen your pencil
答案

这里是“强化练习”，实际上和铅笔有关(哦，同样还有图像)。这个练习包括一些问题：给你一根普通的新铅笔，如果你想用它来画一道实线，用完整支铅笔，线能画多长？

这和图像有什么关系？你必须写些HTML来找到这个答案。这些问题的答案包含在这个URL：http：//www.headfirstlabs.com/trivia/pencil.gif下的图像中。你的工作是给HTML添加一张图像并找到答案：

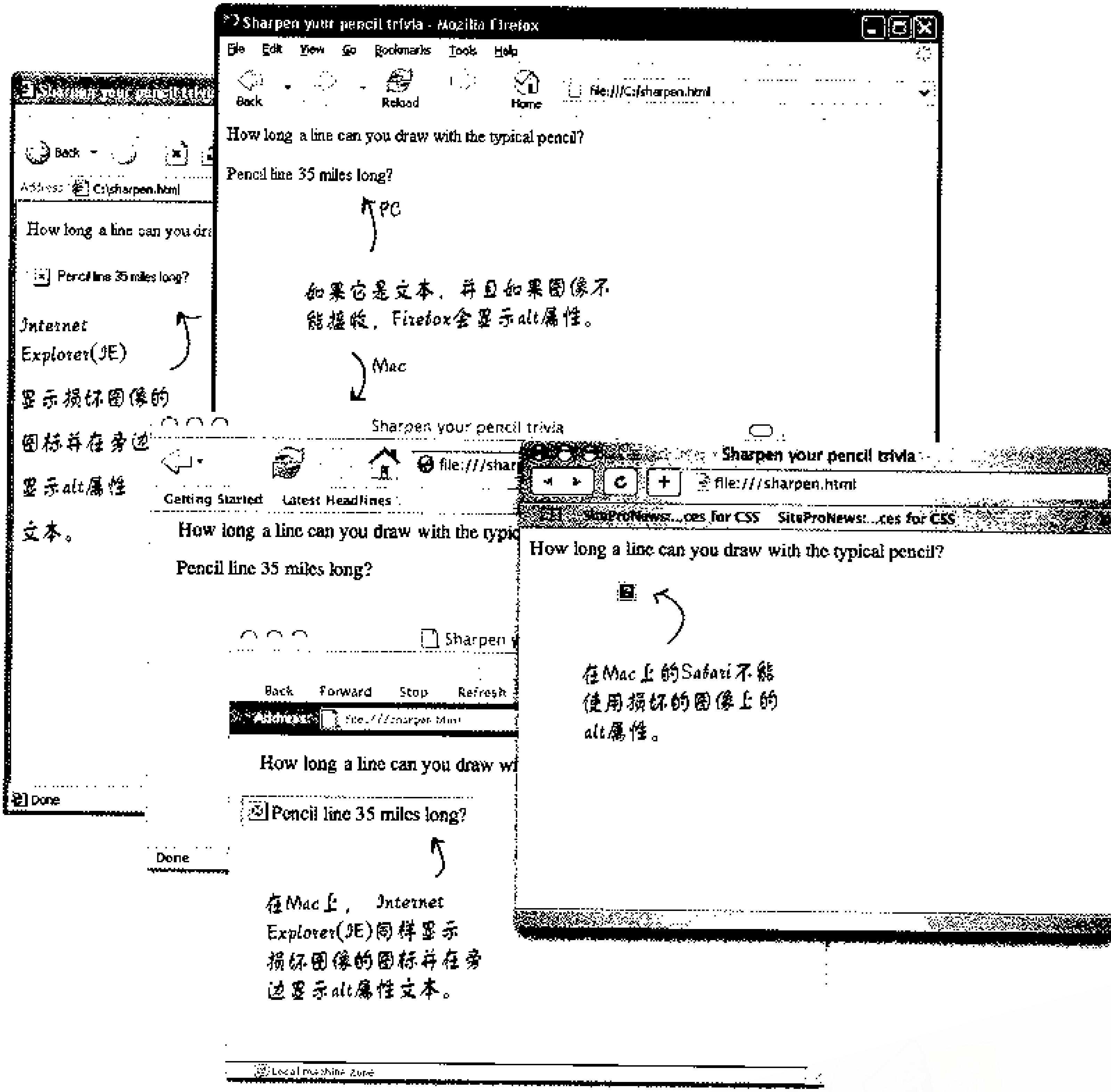
```
<html>
   <head>
      <title>Sharpen your pencil trivia</title>
   </head>
   <body>
      <p>How long a line can you draw with the typical pencil?</p>
      <p>
         <img src="http://www.headfirstlabs.com/trivia/pencil.gif">
      </p>
   </body>
</html>
```



Source: http://www.papermate.com

练习答案

这里是不同浏览器导致的损坏的图像。大多数情况下，浏览器能使用额外的alt属性信息来改进显示。我们要做什么？毕竟，那是网页上的一个错误，我们应该修复它，对不对？好，在现实生活中事物通常不是完美的，有时东西损坏了，网页下载时互联网连接非常差，或者视觉有障碍的用户需要听图像里有什么，因为他们看不到。

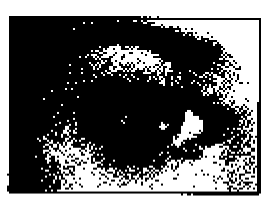


练习答案

用哪种图像格式？

注意你的数字会不同，这取决于你用的软件的版本。

你注意到图像质量是如何递减的吗？

	Format	Quality	Size	Time	Winner
	JPEG	Maximum	232K	83 Seconds	☐
	JPEG	Very High	112K	41 Seconds	☐
	JPEG	High	64K	24 Seconds	☐
	JPEG	Medium	30K	12 Seconds	☑
	JPEG	Low	18K	7 Seconds	☐
	GIF	N/A	221K	80 Seconds	☐

Medium就是优胜者吗？不一定，要视你的需要而定。如果你想要一个高质量图像，你可能要Very High。如果你想要最快的网站，试试Low吧。我们选择Medium因为它是质量和大小的折衷方案。也许你觉得Low就已经足够好，或者值得提升质量到High。所以，这是非常主观的。然而GIF不太适合这类图像，这点是很确定的(这不用你讲)。

练习答案

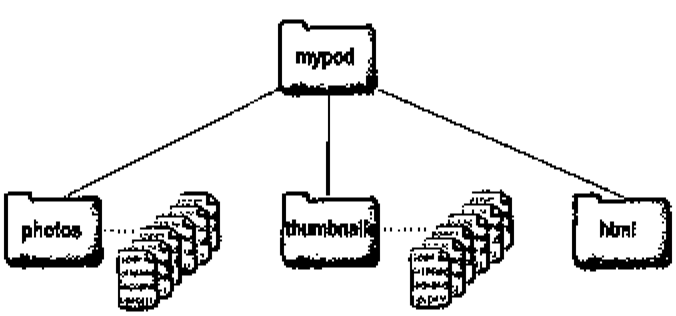
看看本章示例文件中的“html”文件夹，你会发现所有单独图像页面已经存在，除了“seattle_downtown.jpg”页面。在“html”文件夹下创建一个叫“seattle_downtown.html”的页面，并测试它。在你继续操作前先确保它能正常工作。

这里是答案：

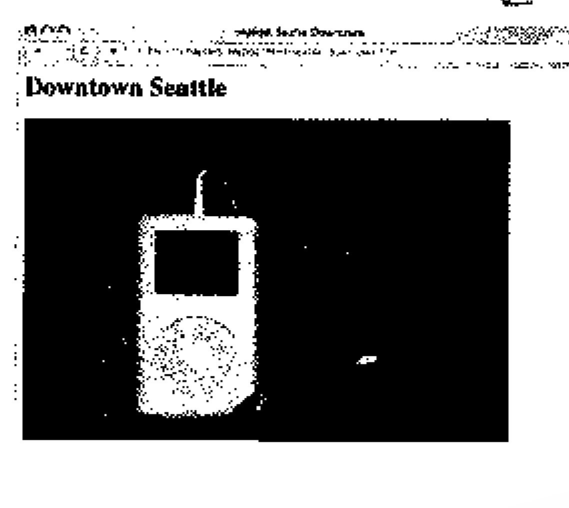
这里是HTML，这个文件应该叫做“seattle_downtown.html”。

```
<html>
   <head>
      <title>myPod: Seattle Downtown</title>
      <style type="text/css"> body { background-color: #eaf3da; } </style>
   </head>
   <body>
      <h1>Downtown Seattle</h1>
      <p>
      <img src="../photos/seattle_downtown.jpg" alt="An iPod in downtown Seattle, WA">
      </p>
   </body>
</html>
```

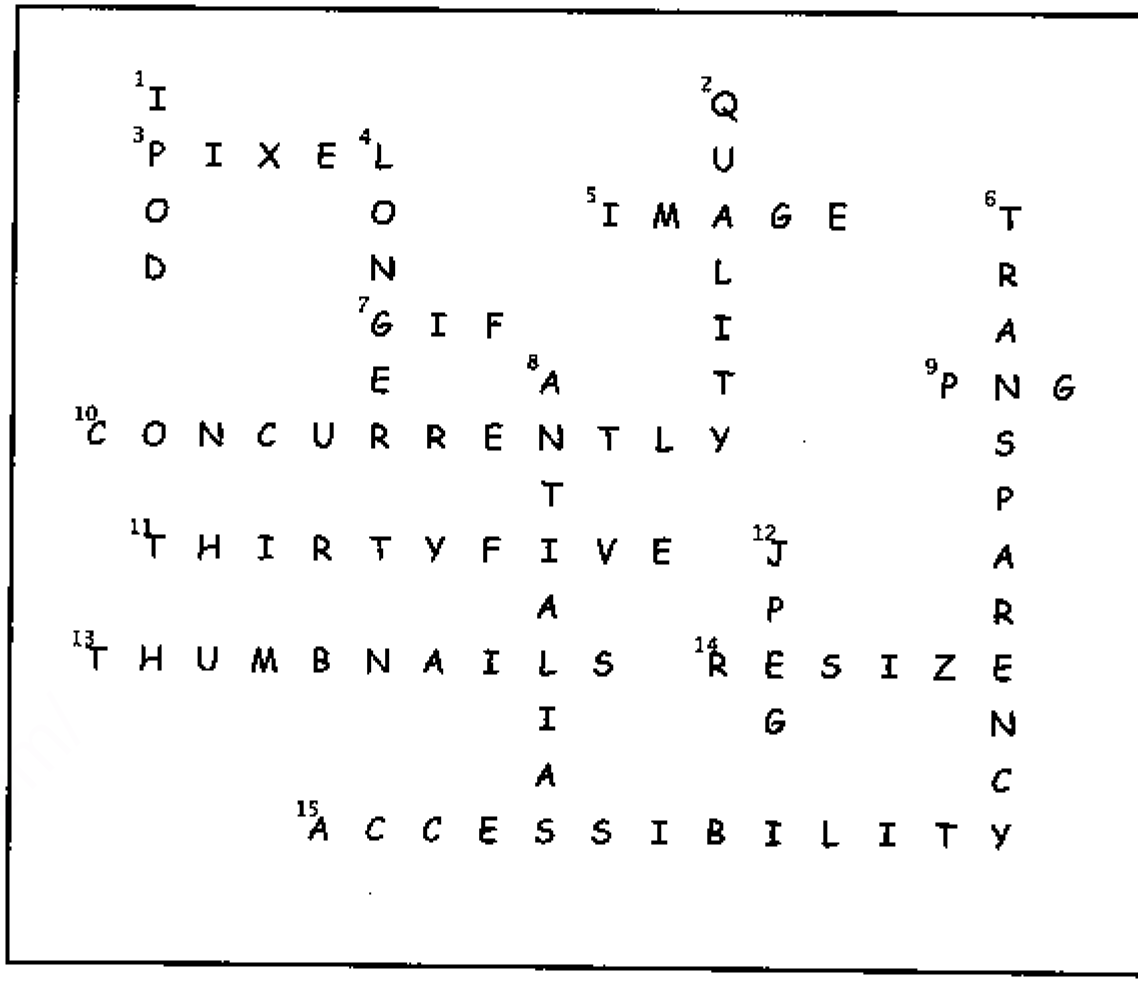
这个文件夹应该在“mypod”下的“html”文件夹中。



这里是测试结果。



练习答案



Sharpen your pencil 答案

这里是添加“seattll.jpg”图像到“index.html”文件的方法。

```
<h2>Seattle, Washington</h2>
<p>
    Me and my iPod in Seattle!  You can see rain clouds and the
    Space Needle.   You can't see the 628 coffee shops.
</p>

<p>
    <img src="photos/seattle.jpg" alt="My iPod in Seattle, WA">
</p>
```

严格的HTML



关于HTML，还有什么知识需要了解呢？ 你已经熟练地掌握了一部分HTML。接下来我们开始学习CSS。通过它，我们可以让所有乏味的标记变得生动有趣。不过，首先要严格规范我们编写HTML的方法，以确保你的HTML结构严谨。别误会，你一直在使用一流的HTML，不过你还可以做点别的让浏览器如实地显示你的网页并确保标记没有错误。这对你有什么好处呢？网页可以在浏览器上得到更统一的显示（甚至在移动设备和为视觉障碍用户提供的屏幕读取器上能更好地显示），网页导入更快，并能保证网页更好地与CSS结合。准备好，在这一章我们将从制作网页的菜鸟转变为高手。



Jim：严格规范？

Frank：就是确保它符合HTML“标准”。

Jim：我们的HTML已经很棒了……这儿，看看它在浏览器上的显示结果，看起来漂亮极了。我们是个好团队，我们知道如何正确地组织我们的元素。

Joe：嗯，我觉得也是……他们只是不想让我们无所事事。什么标准，要求。我们清楚自己在做什么。

Frank：伙计们，其实，虽然我不愿承认，但我还是认为在这个问题上老板是对的。

Jim，Joe：什么？

Joe：看吧，这意味着更多的工作。已经做得够多了。

Frank：伙计们，我的意思是，我觉得这将减少我们以后的工作量。

Joe：哈，听起来不赖……

Frank：好的，问题出来了：浏览器读取我们的HTML然后尽它所能去显示它，是吗？事实上，浏览器是很宽容的……你时不时地犯一些错误，或者使用HTML不够严谨——比如粗心地把一个块元素放到一个内联元素里——而浏览器尽力使网页正确显示。

Jim：哦，还有呢？

Frank：但不同的浏览器（比如是IE，Firefox和Safari）用不同的方法处理不够完美的HTML。也就是说，如果你的HTML有错误，那就别指望你的网页在不同的浏览器中都能很好地显示了。只有没有犯错误，大部分的浏览器才会得到一致的显示结果。而且，当我们开始向HTML添加CSS时，如果我们的HTML不符合标准，这些不同之处将会更加明显。

因此，确保我们的HTML严格符合标准，就不会导致网页无法准确地显示给用户，这也将使我们避免许多问题。

Jim：如果那样可以让我少接点深夜投诉电话，这对我来说倒是个好主意。毕竟，我们的用户使用的浏览器的类型太多了。

Joe：稍等一下，我还不理解。我们现在做的HTML不规范吗？我们的HTML有什么错误？

Frank：也许没错，但我们可以来验证一下。

Joe：比如？

Frank：好了，我们可以从给浏览器提供一点帮助，告诉它我们使用的HTML版本。

Joe：我还不知道我们用的是哪个版本呢。

Frank：哈！这可是个大漏洞。好了，让我们开始吧，先弄清我们使用的HTML版本和如何把版本信息提交给浏览器。此外，我们还必须做点别的，但不用急，没什么大不了的。只要这些完成后，使用CSS就会更加容易了。

BRAIN POWER

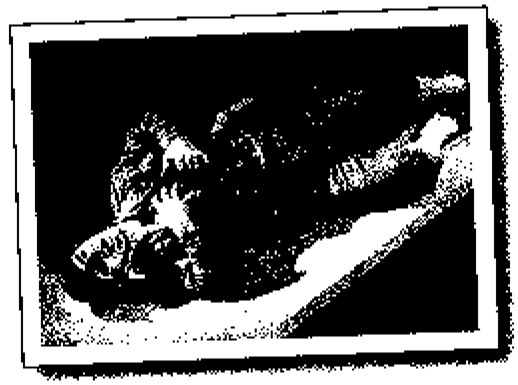
如果你编写的HTML准确无误，浏览器将会一致地显示你的网页，但是如果HTML书写错误或不符合标准，网页在不同的浏览器上显示的结果通常也不同。想想为什么会出现这种情况？

HTML简史



HTML 1.0-2.0

发展初期，关于HTML的认识少之又少。那时的网页很不完善，但至少是超文本的。但没人会去注意表达方面的问题，几乎每个人都有自己的主页。甚至对你书桌上的铅笔、文件夹、笔记本数目的描述都可以作为网页上的内容(你或许会以为我们在开玩笑)。



HTML 3

漫长，惨淡的“浏览器战争”时期。Netscape和微软公司不断推出重大举措试图控制整个领域。因为毕竟谁控制了浏览器部分，就相当于控制了整个领域。

网页开发者是这个战争的焦点。商业战争就像军备竞赛一样，各浏览器商家为了保持领先，不断扩大涉及的范围。谁能胜出呢？事情没那么简单，那时候，你不得不写两份不同的网页：一个用于Netscape浏览器，另一个用于微软的浏览器。这样很不方便。



HTML 4

浏览器大战末期，如我们所愿，世界万维网联盟（俗称W3C）成立了。他们的计划：通过制定统一的HTML标准，使整个产业能有序地发展。

计划的关键是什么？用两种语言分离出HTML的表达式和结构 —— 一种是规范结构的语言（HTML 4.0），一种用于规范表达式（CSS）——并且说服浏览器生产者乐意去采用这些标准。

他们的计划成功实施了吗？

基本上实施了……但也还有些改进（阅读HTML 4.01）。

1989　1991　1995　1998

这一章我们的目标是掌握HTML 4.01。



HTML 4.01

噢，形势大好。THML 4.01作为最先进的版本于1999年问世。人们都希望4.0版本是唯一的标准，以免经常需要更新。

相对于早期的HTML（那段时光就像光着脚在六尺深的雪地中登山的日子），我满足并安心于用4.01版本编写HTML，因为我们知道所有的浏览器都能够很好地显示我们编写的代码。



从第7章开始，我们的目标是专心学好XHTML 1.0。和往常一样，世界不断发展，因此，稍后我们也将讨论这个领域的发展趋势。

XHTML 1.0

但是，在我们都安于现状的时候，新技术出现了，形势也变了。HTML和另一种标识语言——XML合并了，甚至你还来不及给它们"安排婚礼"，XHTML 1.0就诞生了。XHTML 1.0继承了父母的特征：HTML的通用性和与浏览器兼容性，XML的严密性和可扩展性。这意味着什么？你会很快发现，因为在你明白XHTML是"可扩展的超文本标记语言"之前，我们就能让你创建XHTML网页。不过，至少要等到下一章了。

????

将来会发生什么呢？我们会在飞行的汽车里工作，一日三餐吃营养丸？继续阅读去探索答案吧。

1999 2000



公开浏览器的秘密

本周访问：
为什么你会关心你使用的是哪个版本？

Head First：很高兴您能光临，Browser。如你所知，“HTML版本已经成为热门的话题”。为什么会出现这种情况呢？毕竟您是一个Web浏览器。若我给你HTML文件，你就能尽你所能把它显示出来。

Browser：作为一个浏览器，这段时间比较辛苦……大量网页出炉,其中有好多用旧的HTML版本或者标记有错误。如你所说的，我的工作就是尽我所能将那些网页显示出来，不论是什么内容。

Head First：那么最大的问题是什么？和我使用哪一个HTML版本有必然联系吗？

Browser：还记得浏览器大战吗？许多元素加入HTML但我们不支持它们。然而总有些人希望浏览器无论如何也要把它们显示出来，但我们却没办法完全满足他们的要求。

Head First：我想我开始明白问题所在了。那么你如何显示所有这些不同版本的HTML？这是个很高的要求呢。

Browser：嗯，就如我所说的，作为一个浏览器是很辛苦的。解决问题的办法是我们根据两套规则显示网页：一套处理旧版的HTML，一套处理新的，标准的HTML。当我使用旧规则，我称之为“转换显示模式”，因为在这些网页里有很多奇怪的问题出现。

Head First：听起来是一个完美的解决方案……

Browser：然而，那也会给你增添一些麻烦。如果你使用新版的HTML，但是你没有告诉我，那么我就会认为你用的是旧版本，并且进入转换显示模式。但你并不希望这样。

Head First：你的意思是？

Browser：关于显示旧版内容的问题，并非所有的浏览器都达成一致，但对于标准的HTML我们能做出一致的工作。因此，如果你使用标准的HTML，那么把这个信息提交给我们，这样你的设计就可以在所有的浏览器上得到一致的结果了。

Head First：哦，在新版HTML环境下，你就可以摒弃转换显示模式规则了。

Browser：没错。如果不知道你使用的是新版本，我就会进入转换显示模式并尽力去完成工作。但是你不希望如此，因为“转换显示”意味着你的网页显示还会有点异常，只有当我知道你使用的是新版本，你的网页才会显得更美观。

Head First：嗯，那么解决这个问题的方法是什么？我们肯定是希望我们的网页美观点。

Browser：很简单。预先告诉我你使用的是什么版本。那样我就知道该用哪一个规则去显示你的网页了。

Head First：明白了。谢谢您，Brouser。

我不会让你的网页引导浏览器进入转换显示模式！

如果我们预先告诉浏览器："嗨，我是一个完整的网页，我符合标准。我使用的是HTML 4.01。"

这样，浏览器就知道该如何处理你的网页，并且你的网页也会如你所愿很好地被（至少对于你在意的一些浏览器）显示出来。

那么，怎么告诉浏览器呢？很简单，你可在HTML文件的顶部加一行，如下所示语句：

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
```

好了，我们知道这一行冗长难理解，但是，记住了，这是写给浏览器的而不是给你的。这一行叫做文件类型定义。因为它告诉浏览器文件的类型，而在这里，文件就是你的HTML网页。让我们粗略地看一下这一行并简单的了解它。但是我得重申一遍，这是浏览器的职责，你没有必要去记忆或深入了解它。只需要将它置于你的HTML文件的顶端就可以了。

告诉浏览器这是给网页指定一个文件类型。

这表示<HTML>是你的网页的首元素。

这表示 HTML 4.01标准是通用的。

这部分是说明我们使用的是HTML 4.01版本，另外，HTML标记是用英文写的。

你可以把它归为一行，如果你喜欢，你还可以加一个换行。只要把换行置于加引号的各部分之间就可以了。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
      "http://www.w3.org/TR/html4/loose.dtd">
```

注意了，这不是一个HTML元素。开始处，"<"之后有一个"！"，这是特殊之处。

这个指向一个文件，而这个文件用来识别通用的标准。

我们将会对过渡标准多加讨论……

麻烦的是输入的时候还要加斜杠、引号等。所以你可以从文档"doctype.text"拷贝，粘贴这部分，而无需打字输入。当你从headfirstlabs.com站点下载了本书的资源文件，你就可以在"chapter6"文件夹中找到这个文档。

there are no Dumb Questions

问：什么是“规范的”或“标准的”HTML？

答：“标准的HTML”是指HTML使用的版本符合人们认可的“标准”，而现在指的是HTML 4.01。

规范只是你的网页符合标准的另一个说法而已。

问：为什么我应该注意标准的HTML，或者注意让我的网页遵循标准？依我看来它们很不错啊。

答：你真的不在乎编写出来的网页用CSS设计后，最后在一些浏览器上却不能一致（换句话说是显示结果很糟糕）显示的麻烦后果？通过使它们遵循标准，你就可以确保你的网页在不同的浏览器中得到一致的显示结果。

问：那么，我怎样才能保证我的网页遵循标准呢？

答：你必须做一些事,接下来你就会知道，但是我们也会用网上的免费工具来核查你的网页以确保它们是符合标准的。

问：那么，目前我们视HTML 4.01为标准？

答：是的，HTML 4.01是目前最受浏览器欢迎的HTML标准。网络不断向前发展，所以我们将在下一章中讨论什么将是新的标准。

问：如果有HTML 5，那会出现什么情况？

答：问得好。很可能不会出现HTML 5，因为编写网页的新标准是XHTML。下一章你将学到XHTML。你的优势是已经有足够的资本去使用HTML 4.01或XHTML了，不论你选择哪个标准，凭借目前所学的知识，编写网页对你来说已经轻而易举了。

问：如果我在我的HTML文件顶端放置文件类型定义，那么浏览器就会接受它，并且确定我的HTML版本，是吗？

答：没错。文件类型定义告诉你的浏览器：“我使用HTML 4.01”。当浏览器接受了它，它就假定明白你的意思并且知道你使用的是HTML 4.01。这样很好，因为浏览器将有所规划并采用HTML 4.01的规则，而不是采用转换显示模式。

问：如果我告诉浏览器我使用的是HTML 4.01，而实际上并非如此，这样会有什么后果呢？

答：浏览器将会发现你使用的不是HTML 4.01并且回到转换显示模式。这样就会回到原来的问题：不同的浏览器会采用不同的方法处理你的网页。你想得到可预知的结果，唯一方法就是告诉浏览器你使用的是HTML 4.01，并且事实上你也是这么做的。

问：我是不是没有必要去关心文件类型行的内容，只需把它加入我的网页就可以了？

答：是的。不过还有一个问题：你可能想了解几种不同的文件类型，稍后我们将讨论另外一种文件类型。但是，在使用文件类型时，只须把它置于你文件的顶端就可以了。一旦你在那里设置了DOCTYPE，正常情况下没人会去关心它包含什么内容。

问：对于类型定义中的词“transitional”我还不太理解。我觉得它是一个标准，但从字面上看好像离标准还有点差距。

答：把握得很好，你有很好的直觉。如果你继续做一些网页，你自然会弄懂这个问题。



添加文件类型定义

已经讨论得够多了，让我们把DOCTYPE加进HTML。你可以自己输入，也可以从“chapter6”目录下的“doctype.txt”文档里复制并粘贴。

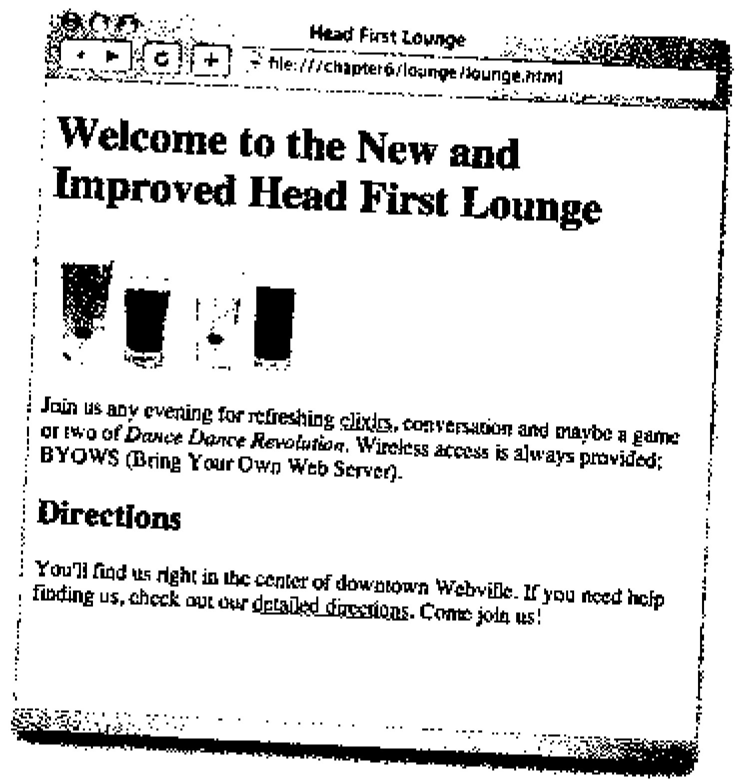
这就是DOCTYPE行。只要把它加到“lounge.html”文件的开头就可以了。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
   "http://www.w3.org/TR/html4/loose.dtd">
<html>
  <head>
    <title>Head First Lounge</title>
  </head>
  <body>
    <h1>Welcome to the New and Improved Head First Lounge</h1>
       <img src="drinks.gif">
    <p>
       Join us any evening for refreshing
       <a href="elixir.html">elixirs</a>,
       conversation and maybe a game or two
       of <em>Dance Dance Revolution</em>.
       Wireless access is always provided;
       BYOWS (Bring Your Own Web Server).
    </p>
    <h2>Directions</h2>
    <p>
       You'll find us right in the center of downtown
       Webville.  If you need help finding us, check out our
       <a href="directions.html">detailed directions</a>.
       Come join us!
    </p>
  </body>
</html>
```

记住，你可以把所有的内容归为一行，也可以像这样，在带引号的部分之间换行。

DOCTYPE测试

改动“chapter6/ lounge”文件夹下的“lounge.html”文档，然后将网页加载到浏览器。



好极了。没有异常之处。除此之外我们也没什么期望了，因为DOCTYPE所要做的就是让浏览器知道你使用的是HTML 4.01。

练习

添加DOCTYPE到“directions.html”文档和“elixir.html”文档。并对它们进行进一步测试。比如“lounge.html”，你不会有什么惊喜（但今晚你可能会睡得香一点）。



Jim：是啊，真的很简单。但是有些地方我还是不太理解：我们在文件的开头放置DOCTYPE来告诉浏览器我们的网页使用的是HTML 4.01，但是却不能保证我们使用的确实是HTML 4.01。我们可能犯了错误。这该怎么办？

Frank：你说得对，只有当你确实是使用HTML 4.01时，你对浏览器的允诺才是有效的。这就是我接下来要介绍的。我们可以使用一种免费的在线设备来校验网页，并获知它是否符合标准。

Jim：真的吗？那设备是怎么工作的？

Frank：这个设备首先查看文档类型，然后核查HTML并确保它正确无误的……比如确定标记名正确，元素嵌套合理，内联元素是包含在块元素里面的等。我们称之为校验器。

Jim：哦，这是免费的？这个设备由谁提供的？

Frank：设定标准的那些人。人们称之为World Wide Web Consortium，或者简称为W3C。

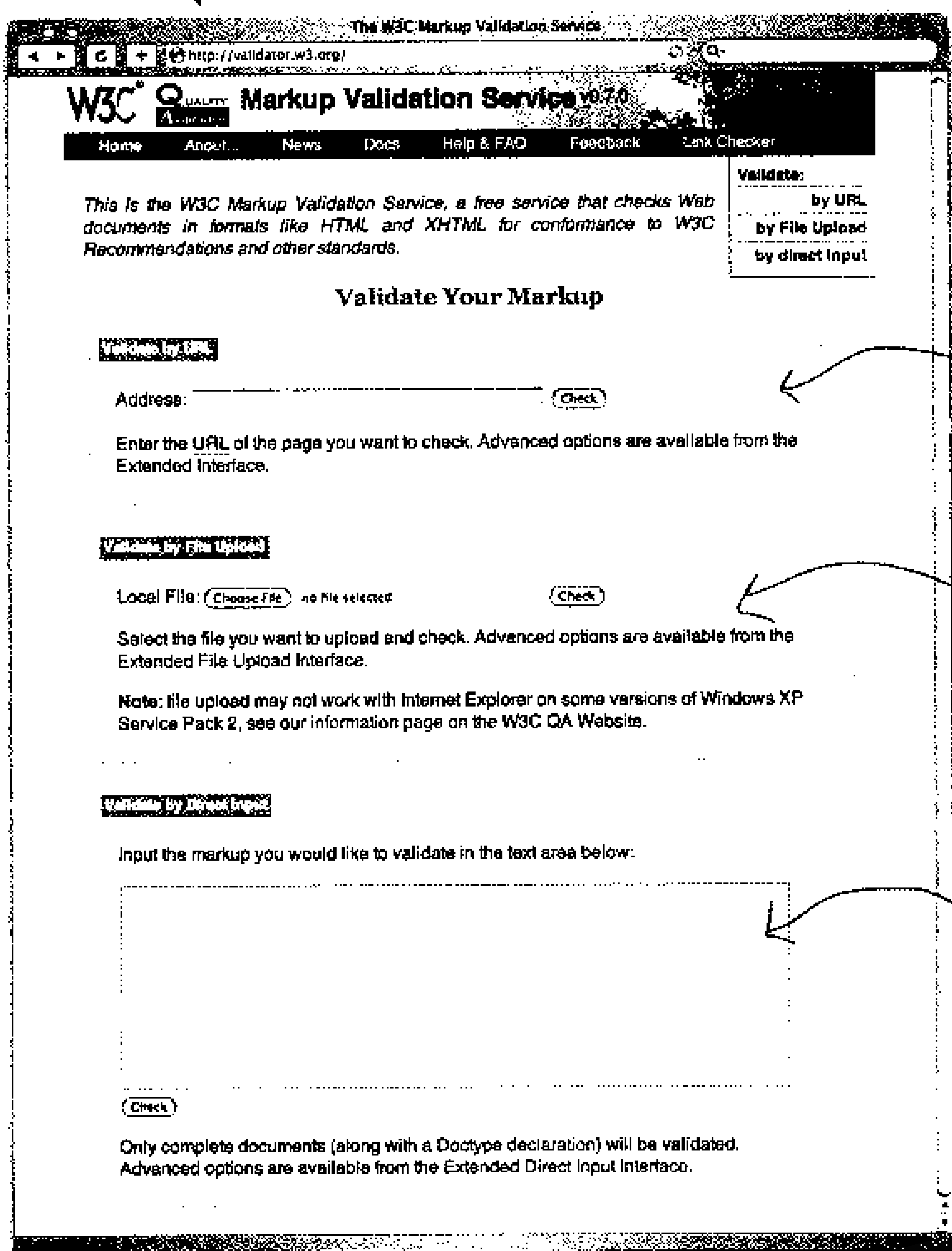
Jim：这些听起来像是编写HTML的关键。但是你刚才提到的东西，比如说什么元素应该包含于什么元素，我怎么才能学到呢？

Frank：让我们先检验一下校验器，然后再来看这个问题……

接触W3C校验器

让我们运行校验器检验lounge文档。让你的浏览器显示http://validator.w3.org站点，按照图示操作。

W3C校验器位于http://validator.w3.org。



你可以用三种方法核查你的HTML。

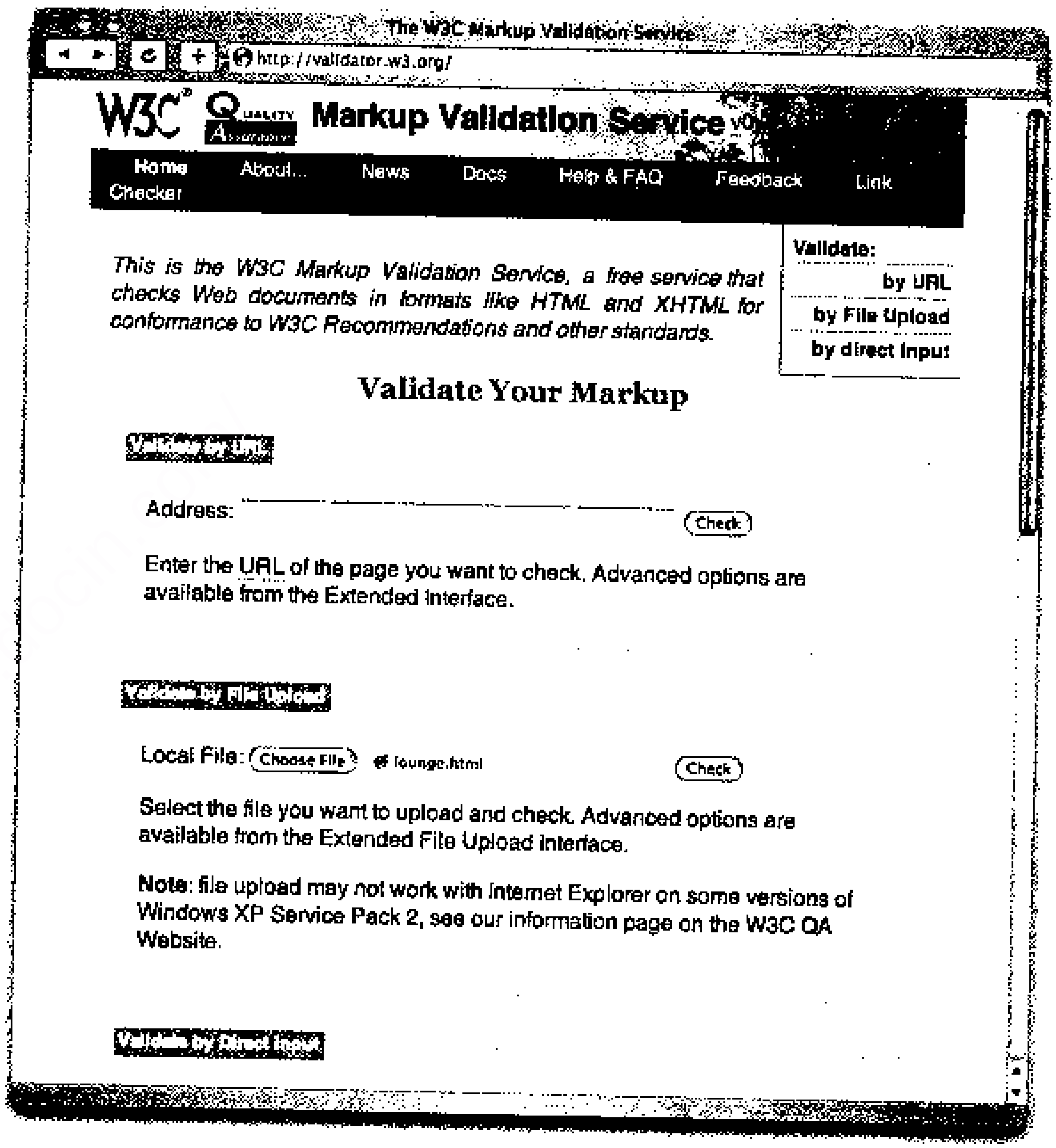
(1) 如果你的网页在互联网上，那么你可以在这里输入网址，然后单击“check”按钮，这样校验器就会去你的HTML并核查它。

(2) 你可以单击“choose file”按钮（或者“browser”按钮——如果你使用的是Windows）并在你的计算机上选择一个HTML文件。选择文件后，单击“check”按钮，然后浏览器就会加载页面给该设备进行核查。

(3) 或者，复制你的HTML文件粘贴在这个表格里，然后单击“check”按钮，该设备随即会对你的HTML进行核查。

校验Head First休闲室

我们将选择方法（3）来检验“lounge.html”文档。也就是说我们需要从“lounge.html”文档中复制HTML再粘贴到W3C校验器网页底部的表格里；按说明进行并试一试……



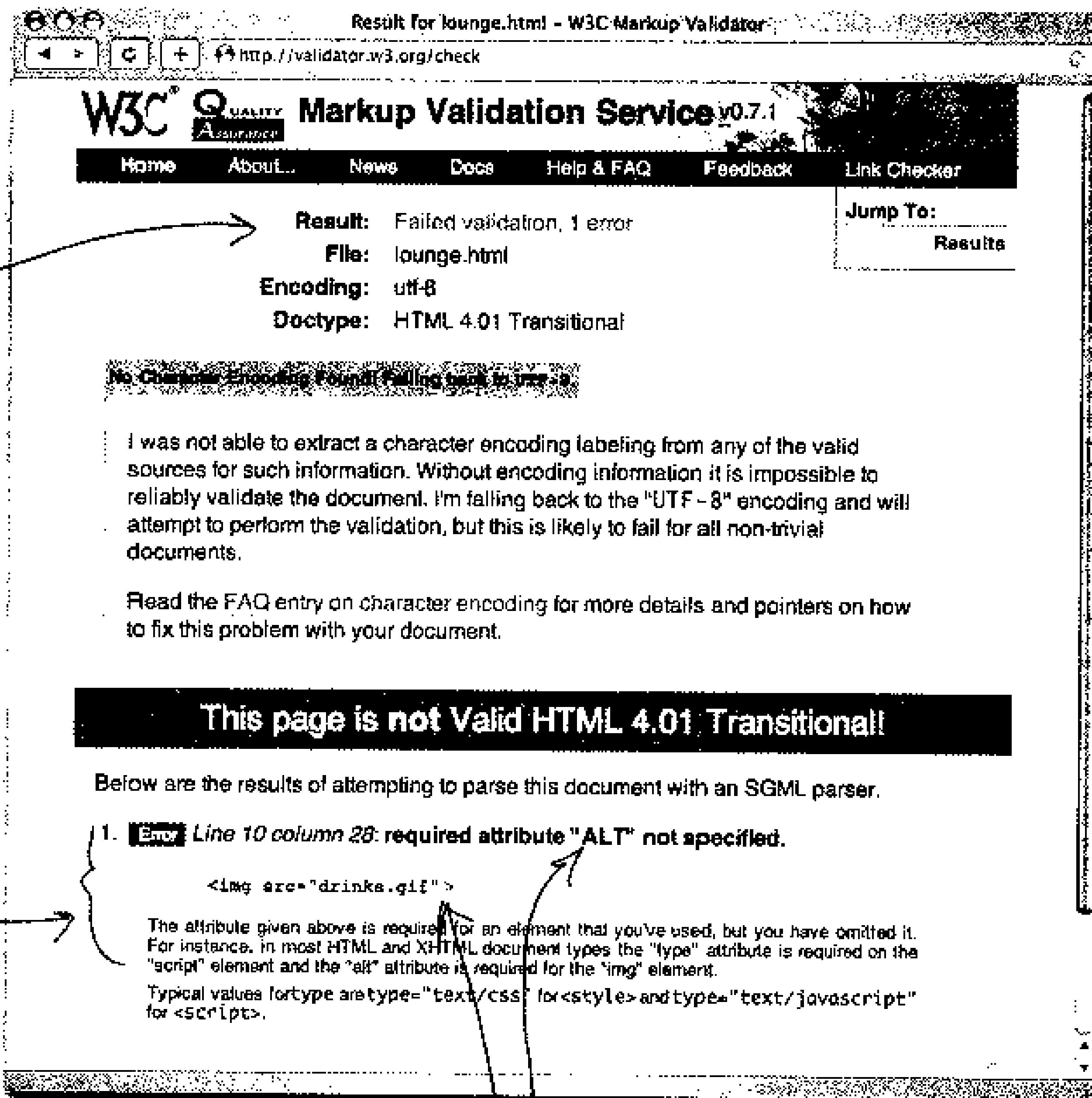
在此，我们使用方法（2）。我们单击“choose file”按钮并且浏览“lounge.html”文档，而这个文档顶部有DOCTYPE来指明使用的是HTML 4.01。我们为这重要的时刻做准备……网页是否会被检验？一切都没有问题吗？单击“check”按钮去查看结果吧……

感觉使用方法（1）或方法（3）比较方便。

Houston，我们遇到问题了……

网页上的那红色提示看起来不妙，好像是我们的网页没有通过校验。我们得研究一下……

校验失败，看起来是有个错误。



这肯定就是错误了吧。

这看起来不难。它的意思是在HTML 4.01中，<img>元素中必须添加一个alt属性。

注意！ W3C正不断地改进校验。

因为W3C不断地改进校验器，所以你可能会看到不同的错误信息。别着急，即使你不看上面的错误，也要按顺序往下阅读，因为下面几页的内容很重要。

解决错误

好的,这个问题看起来很容易解决。你只需在HTML 4.01中把**alt**属性添加到你的**<img>**元素中就行了。继续操作，打开“lounge.html”文档，进行修改，保存，然后重新校验。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
   "http://www.w3.org/TR/html4/loose.dtd">
<html>
  <head>
    <title>Head First Lounge</title>
  </head>
  <body>
    <h1>Welcome to the New and Improved Head First Lounge</h1>
       <img src="drinks.gif" alt="Drinks">
    <p>
       Join us any evening for refreshing
       <a href="elixir.html">elixirs</a>,
       conversation and maybe a game or two
       of <em>Dance Dance Revolution</em>.
       Wireless access is always provided;
       BYOWS (Bring Your Own Web Server).
    </p>
    <h2>Directions</h2>
    <p>
       You'll find us right in the center of downtown
       Webville. If you need help finding us, check out our
       <a href="directions.html">detailed directions</a>.
       Come join us!
    </p>
  </body>
</html>
```

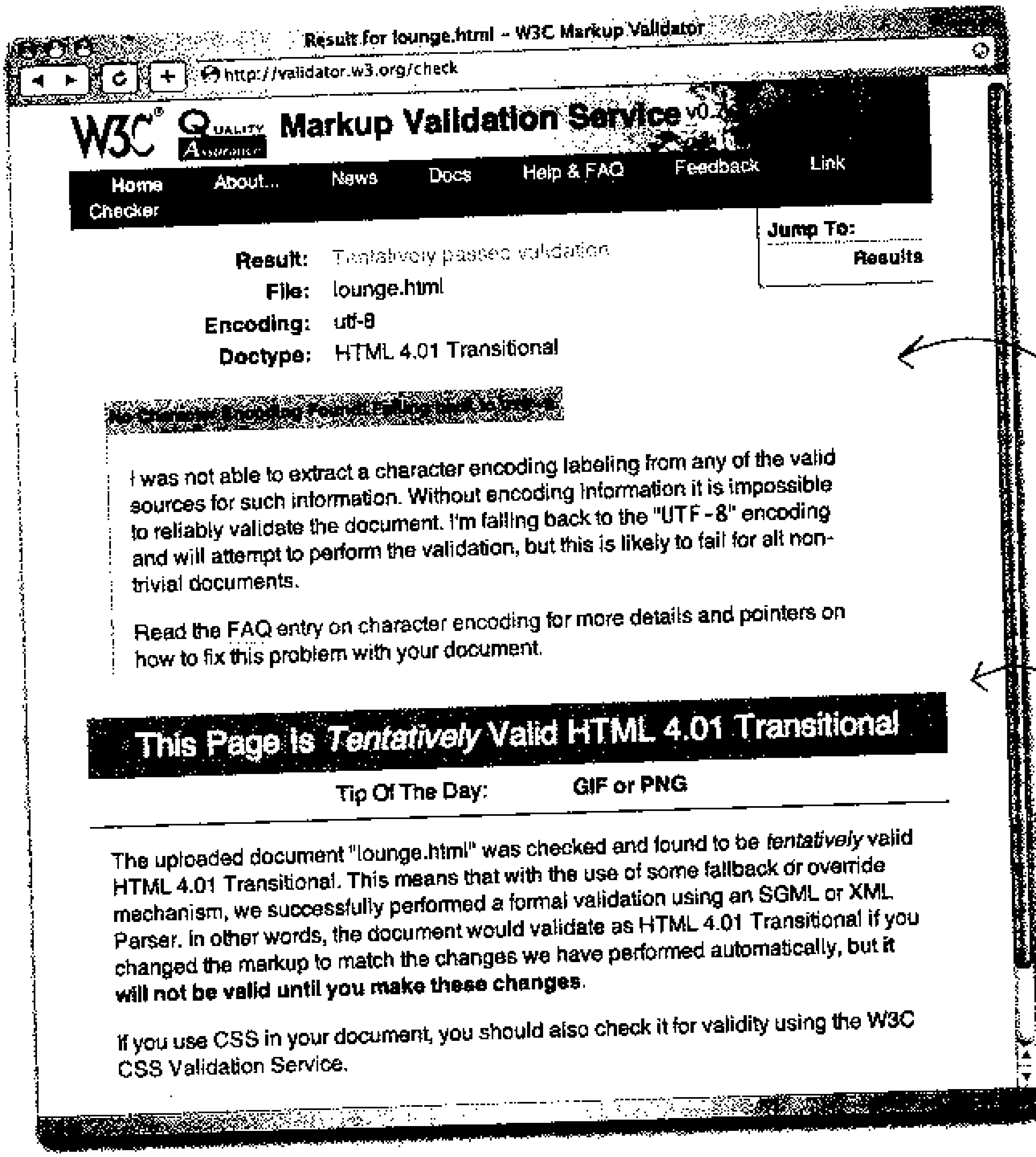
alt属性你是知道的，把它添加到<img>元素里。

BRAIN POWER

为什么在HTML 4.01里**alt**属性是必须的呢？

我们还没达到标准……

嗯，看起来，我们现在达到了“临时合法的HTML 4.01标准”。那听起来就如“很接近了，但还没有完全成功”。那我们看一下：

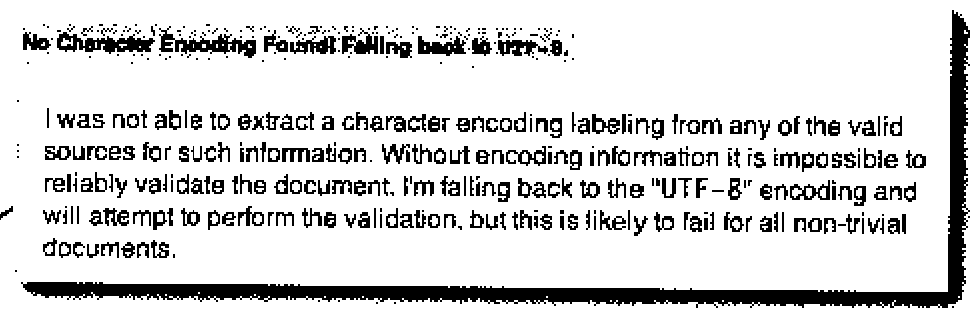


这里确实有几个关于网页的结论，但这究竟意味着什么呢？

这看起来像是我们已经解决了问题，我们的HTML是合法的。

那么，通过检验HTML的书写，我们得到了一个合法的HTML文件。不过看来我们不得不告诉浏览器一些关于我们的“字符编码”的情况。为了解决这个问题我们得查出它指的是什么意思……




No Character Encoding Found! Falling back to UTF-8.

I was not able to extract a character encoding labeling from any of the valid sources for such information. Without encoding information it is impossible to reliably validate the document. I'm falling back to the "UTF-8" encoding and will attempt to perform the validation, but this is likely to fail for all non-trivial documents.

Frank：字符编码告诉浏览器哪类字符用于网页。举个例子，我们可以用英语、中文、阿拉伯语等编写网页代码。

Jim：领会如何显示一个字符的难点是什么。如果文件中有一个“A”，那么浏览器就将显示“A”，对吗？

Frank：如果在网页中我们使用的是中文呢？那是些完全不同的字符而且会有远远超出A~Z26个字符的数量。

Jim：嗯，切中要害了……但是浏览器应能指出不同之处吧？其他的语言看起来与英语并没有相似之处啊。

Frank：不，浏览器只读取数据。它会假设它得到的是英文字符，但如果事实上不是英文字符呢？它会放弃之前的假设。

Jim：这个页面已经打开很长时间了，怎么还没解决这个问题呢？

Frank：因为浏览器正说“嗨，如果要我校验你的网页，你最好预先告诉我你使用的是哪一种字符！”，而我们得满足它的这个要求。别着急，我们只需给网页添加一行，即**<meta>**标记。我本该早点儿想到的。

Jim：对我们还有秘密呀？我还以为在文件中添加了文档类型定义之后，网页就规范了呢。

Frank：但愿不会再有令人费解之处了！让我们添加**<meta>**标记，接着校验吧！

添加<meta>标记说明内容的类型

本书的大部分读者大概都使用英语或西欧语言（即所谓的拉丁文字），因此你们必须在HTML中添加如下**<meta>**标记：

```
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
```

你要把这一行当做你HTML中**<head>**元素的首要组成部分添加进去。这个标记告诉所有的浏览器你的文件的内容类型和你编程用的字符类型。让我们详细看一下**<meta>**标记……

"meta"的意思是我们将告诉浏览器关于网页的一些信息……

这部分告诉浏览器更多关于网页内容类型的信息。

Content属性用于指明内容类型的信息。

这是新的部分，它告诉浏览器我们使用ISO—8859—1类型的字符编码。

```
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
```

正如其他的HTML标记一样，<meta>标记具有属性。

首先我们告诉它这是一个HTML文件。这显得有点多余，因为这一点浏览器已经知道了（在DOCTYPE中我们已经告诉它了）。

注意：整个字符串都是Content属性的值。

there are no Dumb Questions

问： DOCTYPES，<meta>标记……这么多，我必须全部记住才能编写网页吗?

答： 声明DOCTYPE和<meta>标记就像纳税：你必须这么做才合法。从另一角度看待这问题：你已经比99%的网页编写者懂得更多了，这已经相当不错。其他人后来才慢慢学会在HTML中添加DOCTYPE和<meta>标记，并坚持使用。因此，只要你使用了正确的DOCTYPE和<meta>标记，就可以做些更有意义的事了。

问： ISO-8859-1?

答： 在这里它和我们一起工作，就像WD-40：你无需关心它为什么叫这个名字，直接使用就可以了。

ISO-8859-1是代表"Latin-1"的字符编码，而Latin-1可以代表大多数的欧洲语言。如果你使用的是其他语言，可以在http:www.w3.orc/International/o-charset.html上查询编码信息。

让校验器更容易识别<meta>内容标记……

好的，现在，我们要把**<meta>**标记一字不差地输入你的HTML中。首先把它添加到“lounge.html”文档中：

这是<meta>标记。我们把他添加到<title>元素上面的<head>元素中。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
   "http://www.w3.org/TR/html4/loose.dtd">
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
    <title>Head First Lounge</title>
  </head>
  <body>
    <h1>Welcome to the New and Improved Head First Lounge</h1>
       <img src="drinks.gif" alt="Drinks">
    <p>
       Join us any evening for refreshing
       <a href="elixir.html">elixirs</a>,
       conversation and maybe a game or two
       of <em>Dance Dance Revolution</em>.
       Wireless access is always provided;
       BYOWS (Bring Your Own Web Server).
    </p>
    <h2>Directions</h2>
    <p>
       You'll find us right in the center of downtown
       Webville. If you need help finding us, check out our
       <a href="directions.html">detailed directions</a>.
       Come join us!
    </p>
  </body>
</html>
```

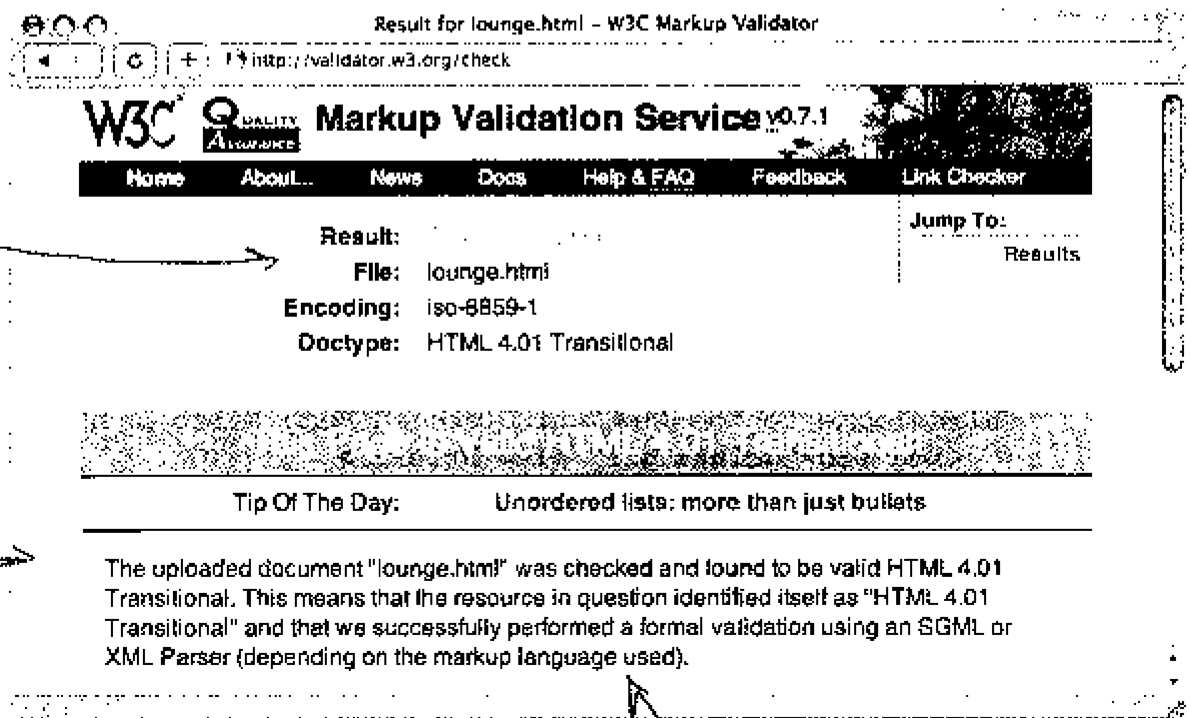
该行总是添加在<head>元素的开头。

还想进行校验？这文件会合法吗？首先，修改“lounge.html”文档，保存后重新载入浏览器。你还是不会发现什么变化，但是浏览器会。现在，让我们看看它能否通过验证……

第三次尝试的魅力

和之前一样，上传你的“lounge.html”文档到http://validator.w3.org上的W3C校验器。或者你可以复制HTML再粘贴到表格中进行校验，甚至可以先把文件移到网站上，再将URL提交给校验器。完成后，就单击“check”按钮！

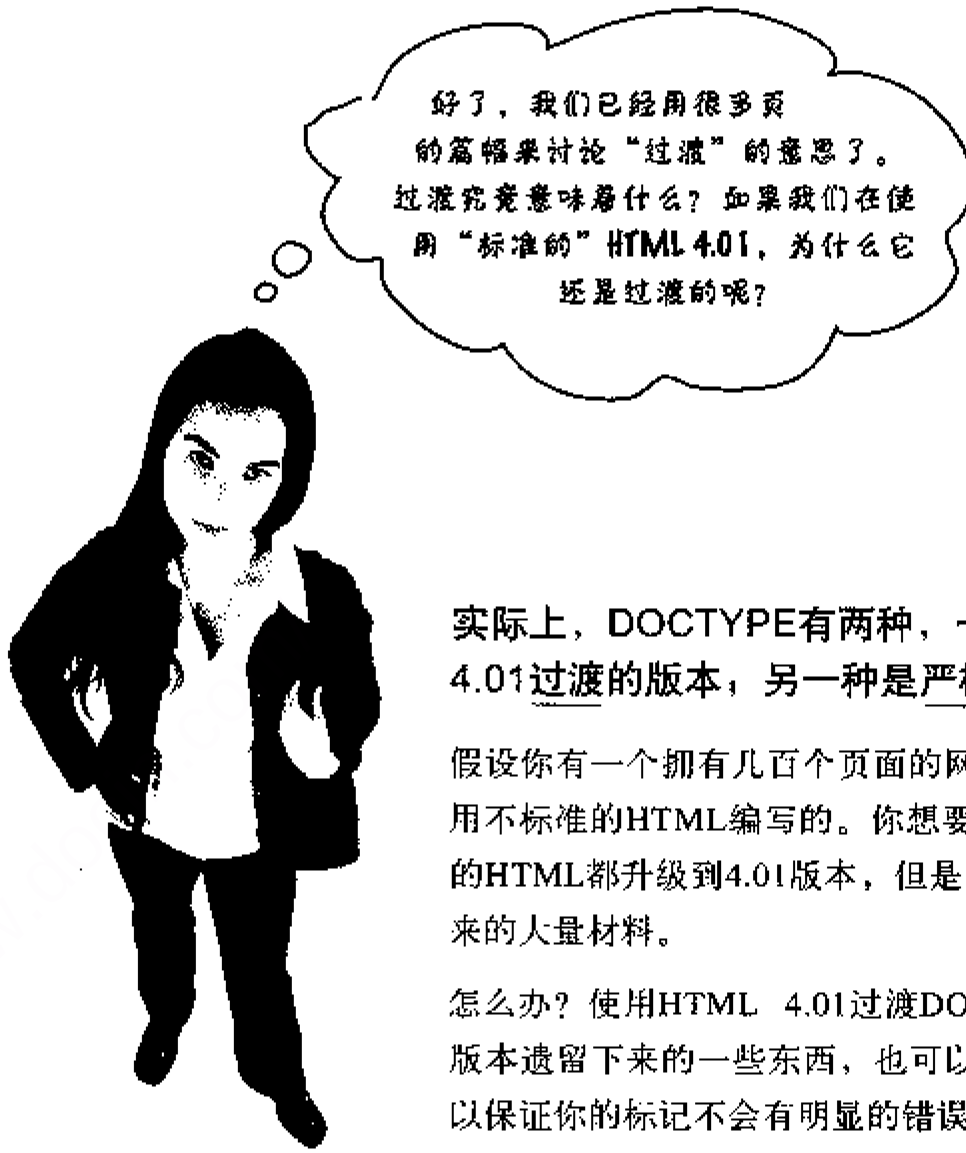
“通过校验”，棒极了！



红色代表不足，绿色代表好的。

如它所说的，你告诉浏览器（和校验器）你使用的是HTML 4.01的过渡标准，而且该设备能成功地校验你的网页。



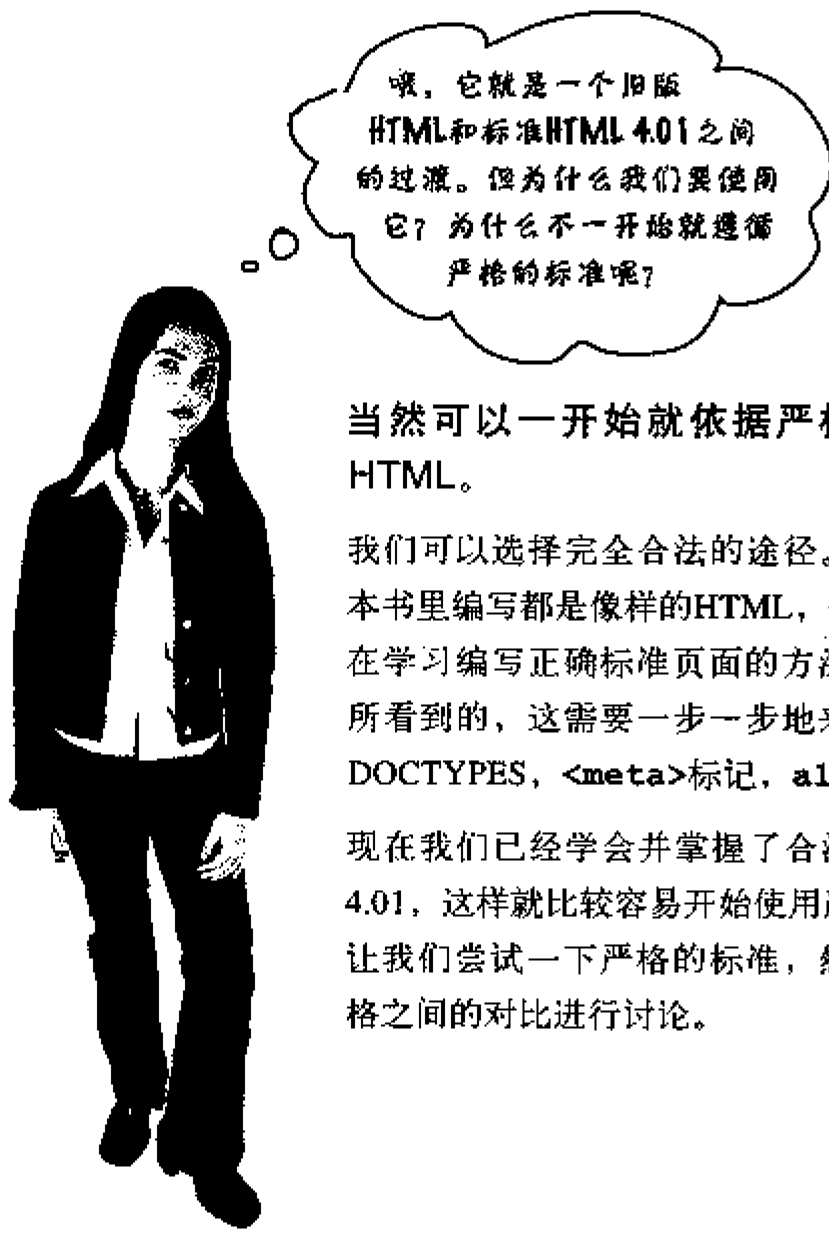


实际上，DOCTYPE有两种，一种是正向HTML 4.01过渡的版本，另一种是严格的HTML 4.01。

假设你有一个拥有几百个页面的网站，而所有的页面都是用不标准的HTML编写的。你想要改进你的站点并把所有的HTML都升级到4.01版本，但是你使用了2.0和3.2遗留下来的大量材料。

怎么办？使用HTML 4.01过渡DOCTYPE，即使你使用旧版本遗留下来的一些东西，也可以通过校验。那样，你可以保证你的标记不会有明显的错误（不会出现如打字错误，匹配错误等），而且不用改写所有的HTML，也能通过校验。

然而，如果你移除所有遗留的HTML，就可以成为严格的文档类型了，你就可以拥有一个完全合法、标准的站点了。



当然可以一开始就依据严格的标准编写HTML。

我们可以选择完全合法的途径。虽然我们在这本书里编写都是像样的HTML，但是我们现在正在学习编写正确标准页面的方法。对吗？如你所看到的，这需要一步一步地来，比如所有的DOCTYPES，**<meta>**标记，**alt**属性等。

现在我们已经学会并掌握了合法的过渡HTML 4.01，这样就比较容易开始使用严格的HTML了。让我们尝试一下严格的标准，然后就过渡与严格之间的对比进行讨论。



为了从过渡的HTML 4.01转化为严格的HTML 4.01，我们把DOCTYPE改为严格的版本。一旦我们这么做了，校验器（和浏览器）就会认为我们遵循了更严格的规则——不允许存在旧版材料的HTML。接下来我们要讨论那些规则，但现在我们应先试一下严格的DOCTYPE。为此，我们得先简单看看DOCTYPE。

这部分与过渡的DOCTYPE相同。

这里没有“Transitional”。

这是HTML 4.01 DOCTYPE的严格版本。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN"
      "http://www.w3.org/TR/html4/strict.dtd">
```

总体上看起来很相似。

记住，这个URL指明了严格的HTML的内容。

没有明显的不同。“Transitional”被移除了并且我们用一个不同的URL指向严格的HTML 4.01版本。让我们用严格的DOCTYPE代替过渡的DOCTYPE，然后进行校验。

把DOCTYPE改为严格版本

再次打开“lounge.html”文档。为了把过渡版本改为严格版本，你必须对DOCTYPE做出修改：删除单词“Transitional”，并在URL中将“loose.dtd”改为“strict.dtd”。如果你喜欢的话，可以删除原来那一行，重新输入新的一行。

首先，去掉“Transitional”。

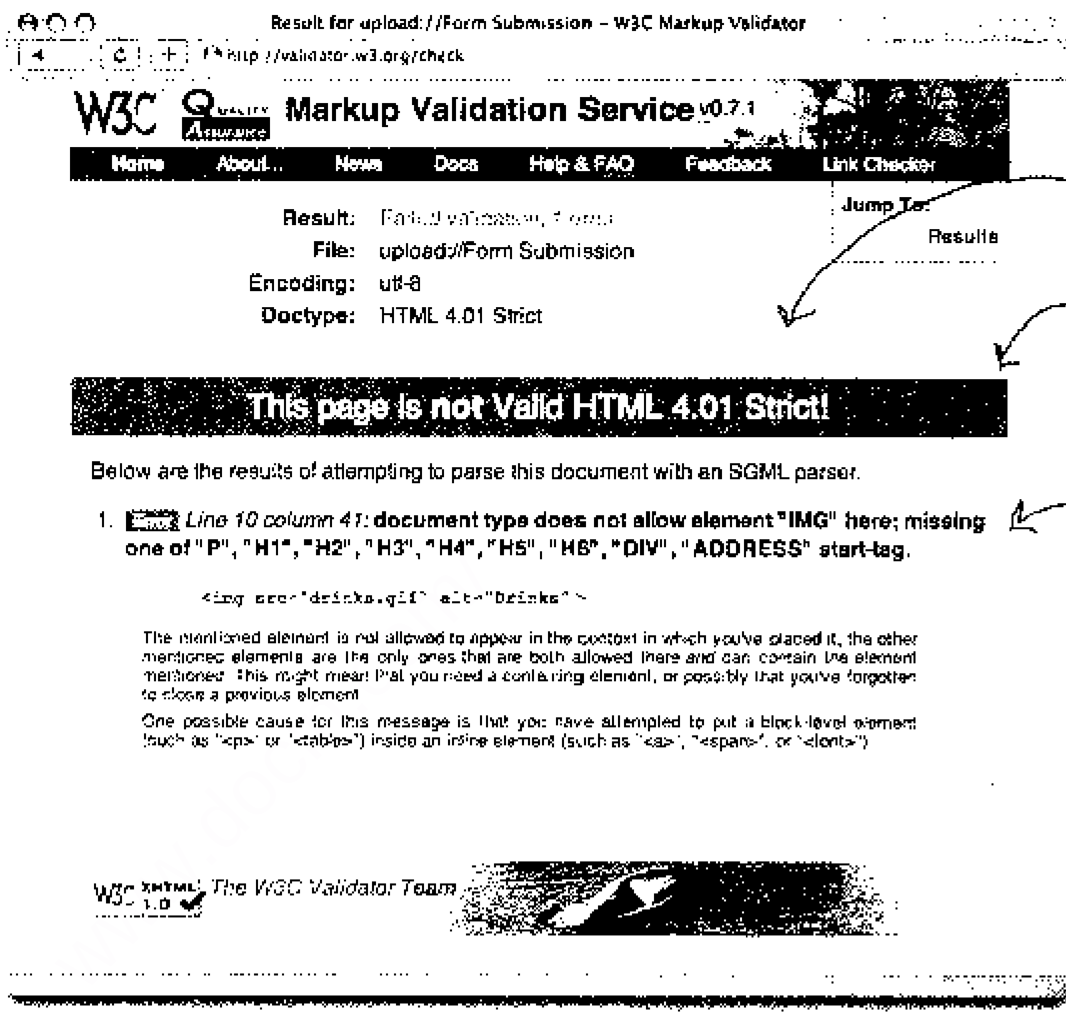
然后，将“loose.dtd”改为“strict.dtd”。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN"
   "http://www.w3.org/TR/html4/strict.dtd">
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
    <title>Head First Lounge</title>
  </head>
  <body>
    <h1>Welcome to the New and Improved Head First Lounge</h1>
       <img src="drinks.gif" alt="Drinks">
    <p>
       Join us any evening for refreshing
       <a href="elixir.html">elixirs</a>,
       conversation and maybe a game or two
       of <em>Dance Dance Revolution</em>.
       Wireless access is always provided;
       BYOWS (Bring Your Own Web Server).
    </p>
    <h2>Directions</h2>
    <p>
       You'll find us right in the center of downtown
       Webville. If you need help finding us, check out our
       <a href="directions.html">detailed directions</a>.
       Come join us!
    </p>
  </body>
</html>
```

就是这样。确定你的DOCTYPE形如上面的例子。

现在，剩下的就是问校验器你的HTML是否符合严格的HTML 4.01版本了。如果确定网页已经修改完毕，那就再次使用校验器检验网页吧。

是否被认可了？

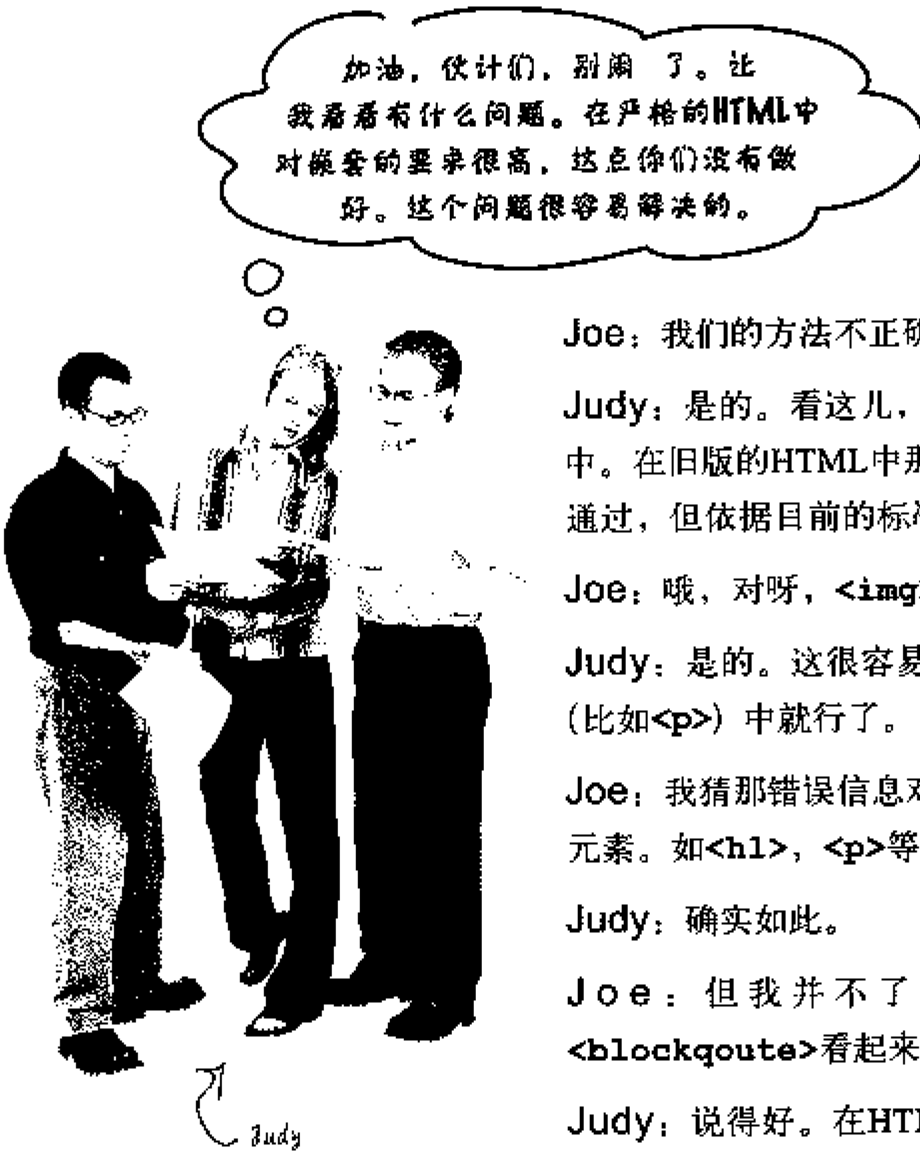


噢，又是红色的，真糟糕！

看来，还不符合严格标准，为什么呢？

让我们看一下错误信息：看来是严格的HTML 4.01不赞同我们放置<img>元素的位置。但过渡的标准不排斥它啊……可能是嵌套规则在严格的HTML中有所更改吧？





Joe：我们的方法不正确？

Judy：是的。看这儿，你把一个<img>元素嵌套到<body>元素中。在旧版的HTML中那是对的，在过渡的HTML 4.01中它也可以通过，但依据目前的标准，内联元素必须包含于块元素。

Joe：哦，对呀，<img>元素是一个内联元素。

Judy：是的。这很容易解决，你们只要把图像粘贴在一个块元素（比如<p>）中就行了。

Joe：我猜那错误信息对你来说不难理解……我知道的元素都是块元素。如<h1>，<p>等。

Judy：确实如此。

Joe：但我并不了解我们使用的所有块元素……比如，<blockqoute>看起来就不在列表上。那是一个块元素，是吗？

Judy：说得好。在HTML 4.01的严格版本里你不能把<img>元素嵌套进任何块元素中。举个例子，你不能把内联元素嵌套在<blockqoute>这个块元素中。

Joe：嗯，在校验之前如何知道嵌套是否合理呢——是否有一份“嵌套规则”列表？

Judy：有，如果你看过那些规则，一般来说就能记住大部分内容了。

Frank：Judy，还有什么地方使用了不合理的嵌套吗？

Judy：我找不出了……我觉得，其他的都很好。因此，我们需要校验器。它们不会漏过任何细微的差错，而人会。

Frank：好的，我们先解决<img>元素，然后开始校验网页。伙计们，我期待看到代表成功的绿条。

Judy：祝你们好运。胜利在望啊。

解决嵌套问题

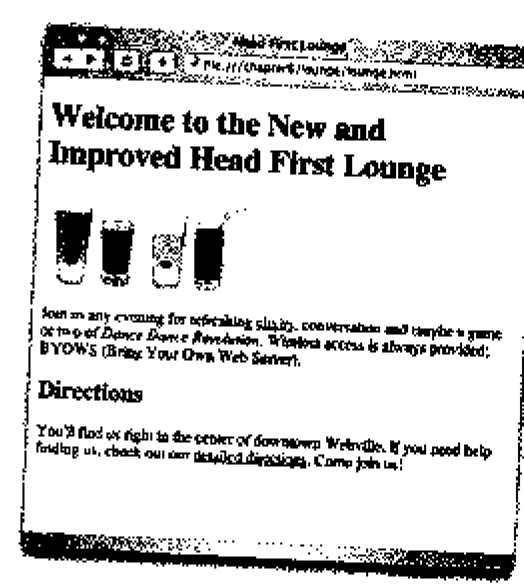
严格的HTML 4.01似乎更喜欢把内联元素如图像，嵌套进块元素（如段落或标题）中。只需做一个简单的修改。打开“lounge.html”文档并在<img>元素旁添加<p>。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN"
   "http://www.w3.org/TR/html4/strict.dtd">
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
    <title>Head First Lounge</title>
  </head>
  <body>
    <h1>Welcome to the New and Improved Head First Lounge</h1>
    <p>
       <img src="drinks.gif" alt="Drinks">
    </p>
    <p>
       Join us any evening for refreshing
       <a href="elixir.html">elixirs</a>,
       conversation and maybe a game or two
       of <em>Dance Dance Revolution</em>.
       Wireless access is always provided;
       BYOWS (Bring Your Own Web Server).
    </p>
    <h2>Directions</h2>
    <p>
       You'll find us right in the center of downtown
       Webville. If you need help finding us, check out our
       <a href="directions.html">detailed directions</a>.
       Come join us!
    </p>
  </body>
</html>
```

现在，图像“drinks.gif”被安全地嵌套到一个<p>元素中。

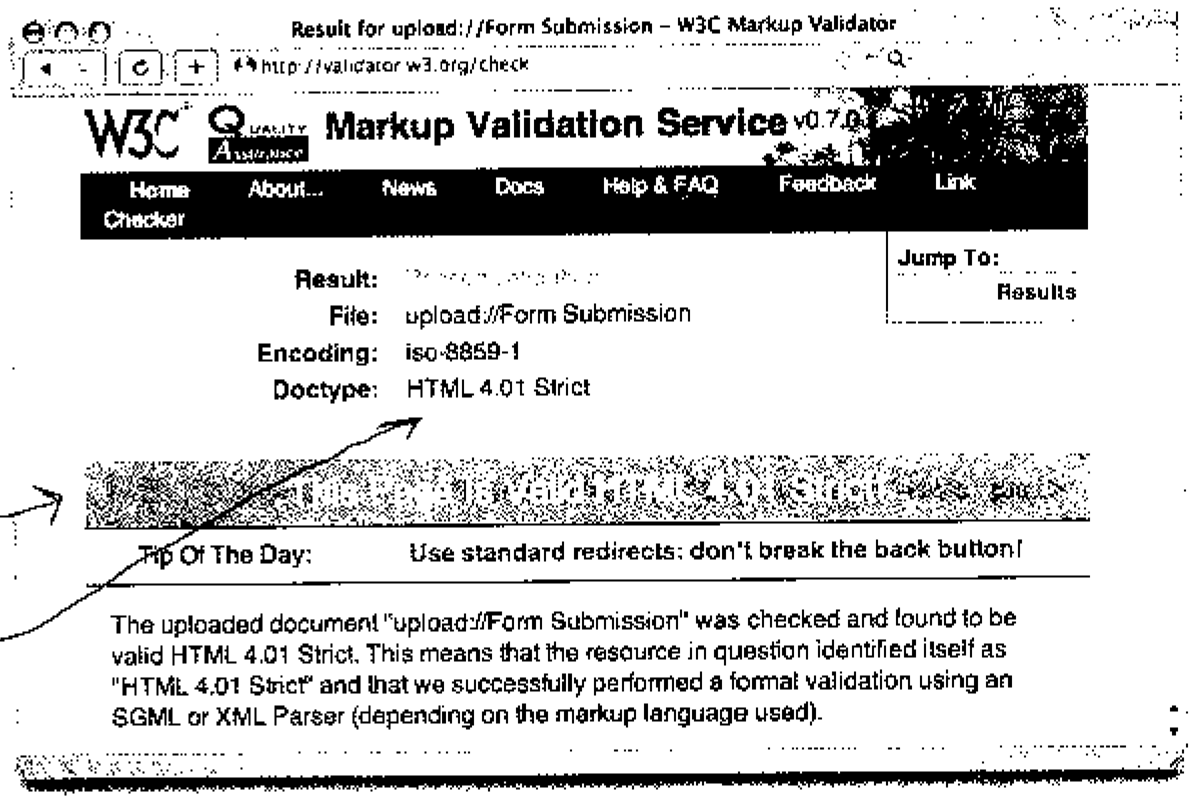
其他所有的内联元素，如<a>和<em>，都已经被嵌套到块元素——这些段落中去了。

完成后保存网页，并重新载入浏览器。你将会发现它并没有真正影响网页的外观。为什么？因为上方的标题和下方的段落是块元素了，而这些块元素的上面和下面分别有换行符。因此，图像旁边的<p>元素实际上没有添加任何换行符或空格。



再一次测试是否严格了……

你知道要做什么。再次要求校验器检验你的“lounge.html”文档，看看我们能否获得认可……



通过了！代表成功的绿色标志。

我们是严格的HTML 4.01。





there are no Dumb Questions

问： **我想我已经掌握了，而且用校验器检验HTML是件有趣的事，但接下来呢？这些所谓的“规范”会有什么实际的好处呢？**

答： 如何取悦用户呢？如果HTML是合法的，网页就能在不同的浏览器中显示一致，这将带给你的用户更好的体验。还有一些其他的好处：标准的网页加载速度更快，并且在其他一些冲浪设备（如电话和电视）上运行地更好。它们更易被因视觉障碍而使用屏幕读取器的用户接受。

问： **可以详细解释先前那错误吗？我想知道它到底是什么意思。**

答： 那错误是由于<img>元素没有嵌套在块元素中造成的。设想一下，浏览器正在读取HTML时，在本应是块元素的地方看到一个<img>元素。它做的第一件事就是说，“嘿，我希望这有一个块元素。”它继续读取，然后移到<img>元素的末尾（因为<img>是个空元素，所以<img>标记末尾的“>”的出现就标志着这个元素的结束）时说，“喂，你不能完成一个<img>元素，因为这里本不该有一个<img>元素”。

此外，你可能会看到由一个错误而引起的多个错误信息。一次考虑一个错误信息，更正错误，那么你可能会发现同时消除了不止一个错误信息。

问： **是不是校验器里所有的错误信息都很难理解？**

答： 大体上是，错误信息可能有点难解释。这是由软件告诉你为什么是错的，而不是人或者Head First的书告诉你。记住，校验器并不知道你下一步将做什么，只能试图指出你的犯错之处并解释错误。它会指出你的错误在哪一行,并且迫切希望你发现错误。

花时间去看那些错误信息，你就会开始了解到它们的窍门。即使校验器没有明确说明，你也能知道它们指的是什么。

问： **为什么错误信息里列出的元素名称是大写的？我还以为元素名称是小写的呢。**

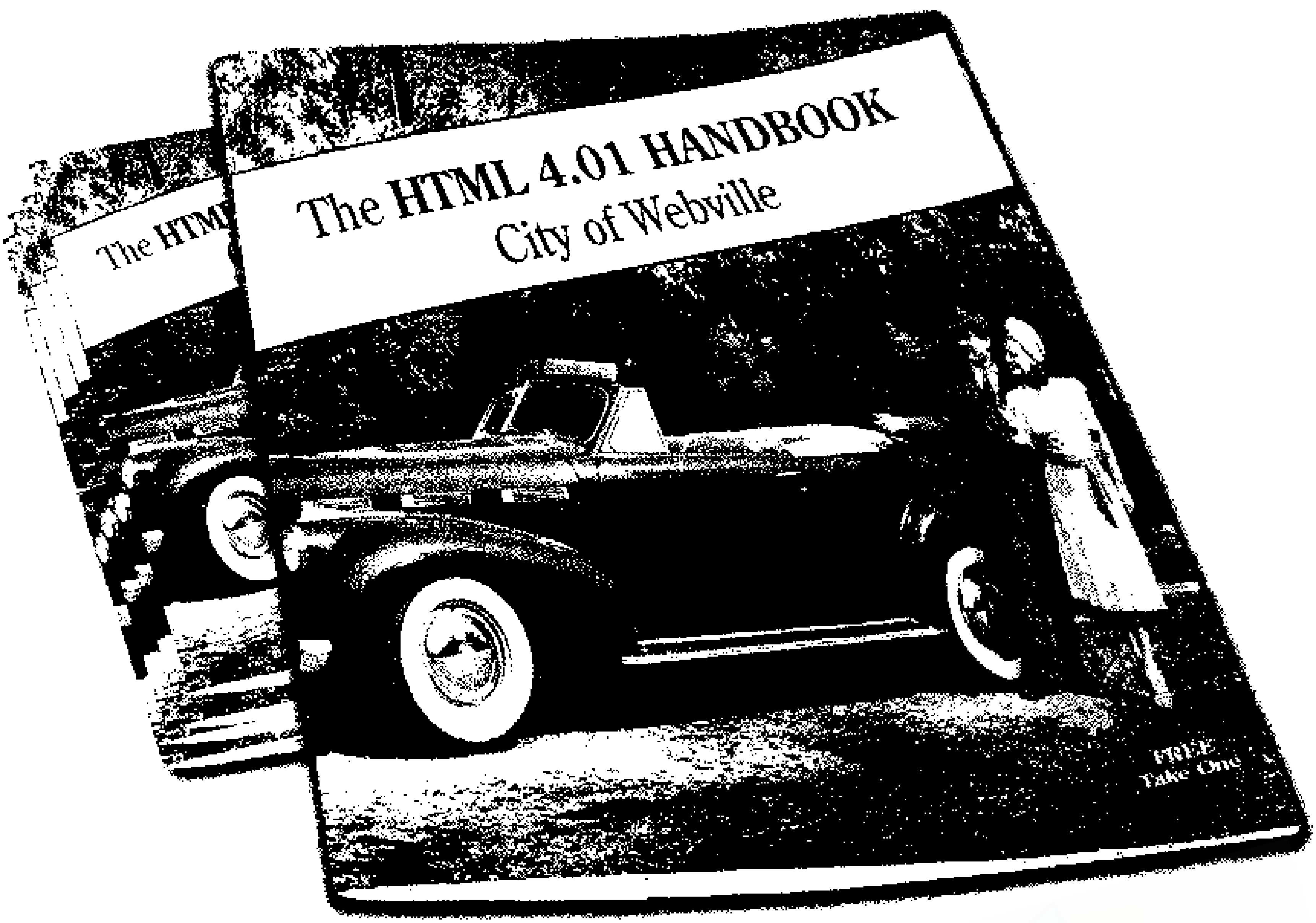
答：问得好。实际上HTML允许元素名称用大写或小写，甚至混合使用也可以。只要你喜欢，你可以写<IMG>或<ImG>。然而，W3C一直在改变规则，因此以后元素名称将会用小写表示。技术上，严格的HTML 4.01的检验器仍然认可（并且显示）大写的标记。我们一直使用小写，所以将会习惯于只使用小写。这意味着接下来你们将不必修改标记名称（也就意味着更少的工作量）。我们所说的“接下来，”实际上是指下一章。

练习

轮到你了。把严格的DOCTYPE和<meta>标记添加到“directions.html”和“elixir.html”文件中。试着校验它们是否合法？如果不合法，修改并重试，直到它们合法。

严格的HTML 4.01，紧扣手册

你已经在Web镇上学习好几章了。该学习HTML 4.01的规则了吧？幸运地，Web镇已经为你使用严格的HTML 4.01准备了便利的手册。这手册是专为你们——初来Web镇的人准备的。它不是详尽的参考书，但集中列出了相对重要的常见规则。有了这个手册，你就能围绕Web镇弄清接下来几章中的规则，能学到很多知识而且它还是免费的。



使用严格的HTML 4.01 Web镇手册

如果你不知道规则，驰骋于信息高速公路将十分危险。在这本简明的手册中，我们从严格的HTML 4.01中摘录了一套常见的规则。首先从这些主要的规则开始。



<html>元素：必不可少的元素之一。

网页总是以一个DOCTYPE开始，紧接着，**<html>**元素必须出现在网页的开头和结尾。因此，在DOCTYPE之后，网页以**<html>**标记开始，以**</html>**标记结束，而网页的其他东西都包含在里面。



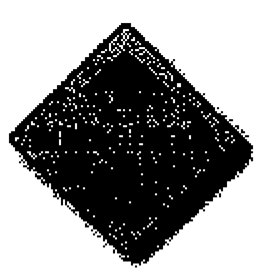
记得使用<head>和<body>让你的网页更好。

只有**<head>**元素和**<body>**元素可以直接包含于**<html>**元素，也就是其他任何元素都必须毫无例外地包含于**<head>**元素或**<body>**元素。



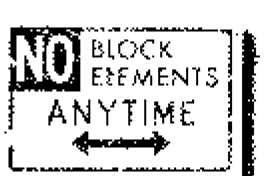
<head>元素里没有<title>元素会怎么样?

我们总是给**<head>**元素一个**<title>**元素，这是规则。不这么做将导致HTML不合法。**<head>**元素是可以放置**<title>**元素，**<meta>**元素和**<style>**元素的唯一地方。



只使用有用的块元素填充<body>。

只能在**<body>**元素里直接放置块元素（**<h1>**,**<h2>**,...,**<h6>**,**<p>**,**<blockqoute>**等）。所有的内联元素和文本都必须在块元素中才能运行。

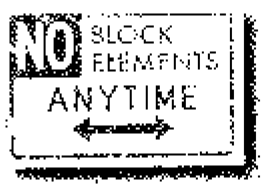


让块元素远离内联元素。

只有文本和其他内联元素可以嵌入内联元素中。块元素在任何情况下都不允许包含在内联元素里。

使用严格的HTML 4.01 Web镇手册续

了解完主要的规则，我们再来看一些细节。



块元素禁止包含在<p>元素之中。

只有文本才能组成段落，所以块元素不允许包含在<p>元素中。当然，只要你喜欢，可以在段落里使用所有的内联元素（**<em>**,**<a>**,**<strong>**,**<img>**,**<q>**,等）。



列表只能包括列表项目。

只有**<li>**元素允许放在**<ul>**元素和**<ol>**元素里。因此，只有列表项才能放在有序或无续列表里。



列表项可以是任何内容。

Web镇对于**<li>**元素的规则很宽松：你可以添加文本、内联元素或块元素到列表项里面。

哦，第 3 章我们做Tiny的<blockquote>时没有遵循标准。那个文本应该先被放入<p>。



谁知道？<blockqoute>只喜欢块元素。

<blockqoute>元素中要有一个或多个块元素。文本直接包含于块引用很常见，但在Web镇中这种做法不合法。请先把文本或内联元素置于块元素之内，再一起添加到**<blockqoute>**。



内联元素相互嵌套时要小心。

可以将任意内联元素嵌入到另一个内联元素中，但有两种情况例外。**<a>**元素决不能自我嵌套，因为那样用户不易理解。另外，据规定空元素（如**<img>**）中不能嵌套其他内联元素。

there are no Dumb Questions

问： **情况不太糟，希望我能记住这些规则。真的只要遵循那些规则就能编写严格的HTML 4.01 吗？**

答： 这些规则很有用，但是记住了，你还没有学会关于HTML的所有东西，还有一些新的东西也是你必须学的。没有必要记住全部规则。多做练习是最好的巩固方式。当你陷入疑惑时，可以查找HTML参考书或询问校验器HTML是否合法（无论如何你必须做到合法）。

问： **无论何时，都要依据严格规则进行吗？**

答： 看情形。如果发布的网页只给三个人看呢？嘿，只要在你们的浏览器上都能很好地显示，谁会在乎其他的呢！但如果要做一个浏览人数相当多的网页，你最好使HTML符合标准并且合法。那应该遵循过渡的标准还是严格的标准呢？该领域在不断地向严格方向发展，因此，你考虑现在的同时也要考虑以后，但最后，你最感兴趣的将会是严格的标准。如果你刚开始入门，遵循严格的标准很简单。如果你使用了严格的标准，那要转向XHTML会更容易，下一章我们将开始学习XHTML，并坚持使用它。

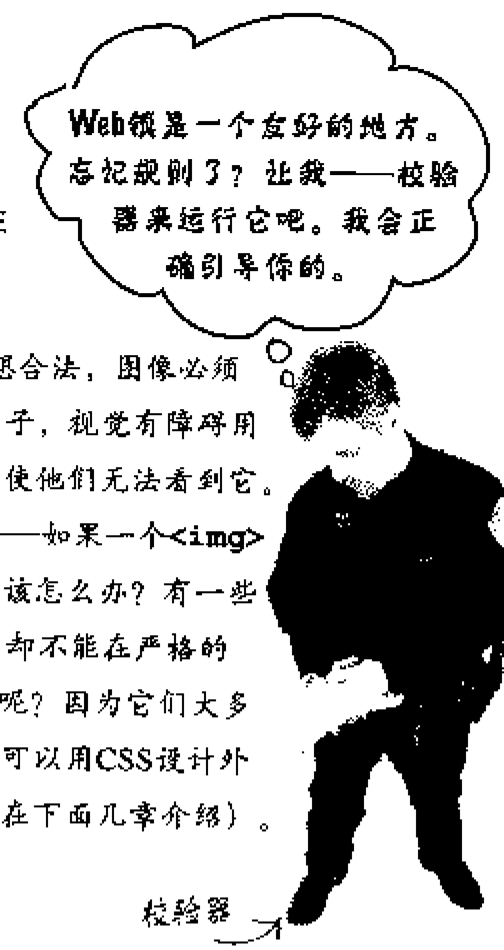
问： **把一个<a>嵌入到另一个<a>会引起混淆，并且不起作用。那我可以把一个<em>嵌入到另一个<em>里吗？这样会有什么后果？**

答： 原则上，可以将emphasis放入emphasis。虽然那看起来很愚蠢，但是既然它不引起问题，就像嵌入一个<a>元素一样，标准只是说，随你的意愿。那<q>嵌入<q>呢，也可以吗？是的，你可以引用其他人引用的东西。所以，大体上，可以嵌套任何内联元素到其他的内联元素。除了<a>元素，其他的内联元素都可以进行自我嵌套。此外还须记住，<img>元素是空的，不能在里面嵌套任何东西。

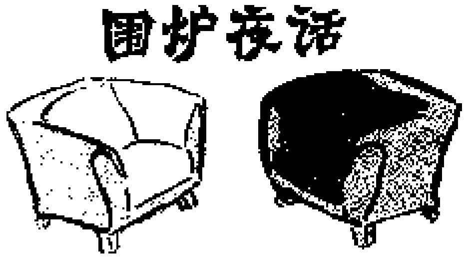
问： **为什么不能直接将文本放入一个<blockqoute>呢？列表项可以包含文本和块元素。这样似乎有点矛盾。**

答： 因为标准就是这么说的。开个玩笑。没错，似乎有点矛盾，但它都是立足于元素的目的。以<p>元素为例。它代表一个文本段落，所以其他的块元素当然不能在它之内。<blockqoute>？它是为了从其他资源引用大段文本，可能还包括标题，段落和其他的部分。所以关键是“块引用”列表项？它们就像元素世界的异类——它们要能够拥有简单的文本，较大的文本（如段落），甚至是其他列表，它们几乎能处理一切问题。

问： **我注意到校验器说标准要求<img>元素必须包含“alt”属性。还有其他的属性是必需的吗？**



答： 问得好。是的，要想合法，图像必须有“alt”属性，所以，举个例子，视觉有障碍用户也可以知道图像的内容，即使他们无法看到它。其他必需的是图像的src属性——如果一个<img>元素指向的却不是图像，那么该怎么办？有一些属性是被HTML3.2认可，但你却不能在严格的HTML4.01 中使用了，为什么呢？因为它们大多会影响网页的视觉效果，但你可以用CSS设计外观（关于这个主题的更多内容在下面几章介绍）。



今夜话题：过渡标准和严格标准试图征召支持者。

过渡标准：喂，严格标准，你到这儿是为了谈论你有多喜欢让那些网页开发者垂头丧气吗？

严格标准：什么意思？

过渡标准：哦，你知道的，所有网页编写者都尽力使网页符合你的严格的DOCTYPE。你知道吗？你太严格了。

严格标准：那是严格的爱，先生。

过渡标准：严格的爱？

严格标准：是的，迟早，任何重要的网页都要追求严格。现在你也许会觉得我过于严格，但到时你会爱我的。

过渡标准：哦，得了吧。没人喜欢时时刻刻都那么严格的。

严格标准：哼！你鼓励人们安于现状，使用那些旧的标记标和属性。你这是在纵容他们。

过渡标准：你知道的，不是每个人都办得到或者愿意花上整夜时间去把整个站点转换为严格的标准。而我，在这当中发挥着重要的作用。

严格标准：这个我明白，人们说他们是“标准HTML”时，事实上，他们还在依赖旧习惯。我想说的是，严格的标准才是正确的选择。它是升级一个站点的唯一方法。

过渡标准：怎么升级某些东西？

严格标准：嘿，先生，你的一些标记已经被“淘汰”了。你知道那是什么意思吗？意味着它们正在逐渐退出舞台。使用严格的标准吧。它会让你更容易更新到下个HTML版本。

过渡标准

所以你就想抛弃数百万仍然使用旧版HTML的网页？完全忽视它们？我猜连你自己都使用了不严格的网页。要不要我取回你的历史清单核查一下？

你知道，不是每个人都愿意站在浪尖上的。有些人喜欢使用那些旧的标记。有些人喜欢调慢进度，确定他们已经了解新的标准后才会开始无奈地着手修改他们的网页。

你真的该优待我；我已经帮助很多人学会遵循严格的标准了。

是的。他们可以从严格的标准开始学，而不需要我。我还是回去专注于我友好温和的从过渡到严格的方法吧。你可以继续你的严格手段。

是，是。你是否也烦扰于在章节末尾告诉读者你也将被淘汰？

严格标准

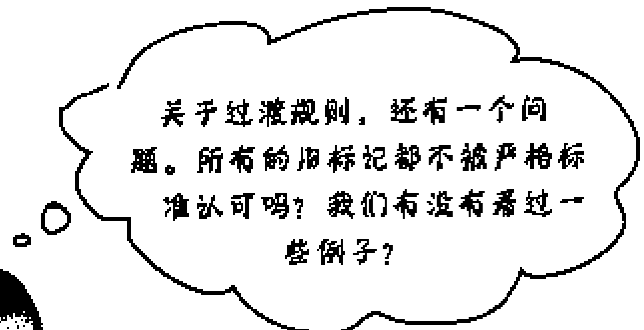
哦，把你的脏手拿开，别靠近我浏览器的历史记录。你说的没错，有许多有用的网页必须更新，并且它们可能不会去更新，但我们应该努力去创建更好的网站。所以，别再鼓励人们安于现状了。

无奈？对于4.01没有什么无奈的。它确实比旧的HTML版本更清晰，更易于理解。况且，如果人们正确地编写网页，那他们就为网页长时间正常运行打下了很好的基础。

让网页符合过渡标准直到学会新的标准，这样确实对人们有帮助。但我想说过渡标准不应该被当成拐杖来用。并且学这本书的人刚接触HTML和CSS，根本就没有必要过渡。

嘿，等着瞧吧。将来网页会感谢我使它们符合严格标准的。

哈哈……





没有，我们编写的严格的HTML标准几乎全部是顺应发展趋势的。

尽管我们已经不再包含DOCTYPE或**<meta>**标记，但是在图像嵌套规则方面我们显得有点懒惰，整本书里你写的HTML很接近标准。因此，你没有多少机会看到已经被淘汰的元素和属性。

这是好事，因为有时候，杜绝坏习惯是学习新标准中最困难的部分。

想看哪些？只需用浏览器访问一些网页并从“查看”菜单中选择“查看源文件”（你的浏览器菜单可能不同）。任何用于改变网页显示的标记或属性都可能被HTML 4.01所排斥（因为现在这是CSS的工作）。不过了解一点那些遗留下来的元素于己无害，因为你很可能会不时地碰上他们。让我们快速看一下……



HTML 考古



我们搜索到一个HTML 3.2网页，这个网页包含了一些不再适用于标准规则的元素和属性，还有一些不再被严格的HTML 4.01 允许的错误。

```
<html>
<head>
   <title>Webville Forecast</title>
</head>

<body bgcolor="tan" text="black">

   <p>
      The weather report says lots of rain and wind in store for
      <font face="arial">Webville</font> today, so be sure to
      stay inside if you can.
   </p>

   <ul>
      <li>Tuesday: Rain and 60 degrees.
      <li>Wednesday: Rain and 62 degrees.
   </ul>

   <p align=right>
     Bring your umbrella!

   <center><font size="small">This page brought to you buy Lou's
       Diner, a Webville institution for over 50 years.
   </font></center>

</body>
</html>
```

这些是控制外观的属性。Bgcolor设置网页的颜色，而text用来设置文本的颜色。

字体的改变由<font>元素和它的face属性来完成的。

没有结束标记（如
和</p>）。

甚至属性值旁边可以没有双引号。

文本大小由<font>元素的size属性控制。

有两种方法用来排列文本——右对齐或居中。

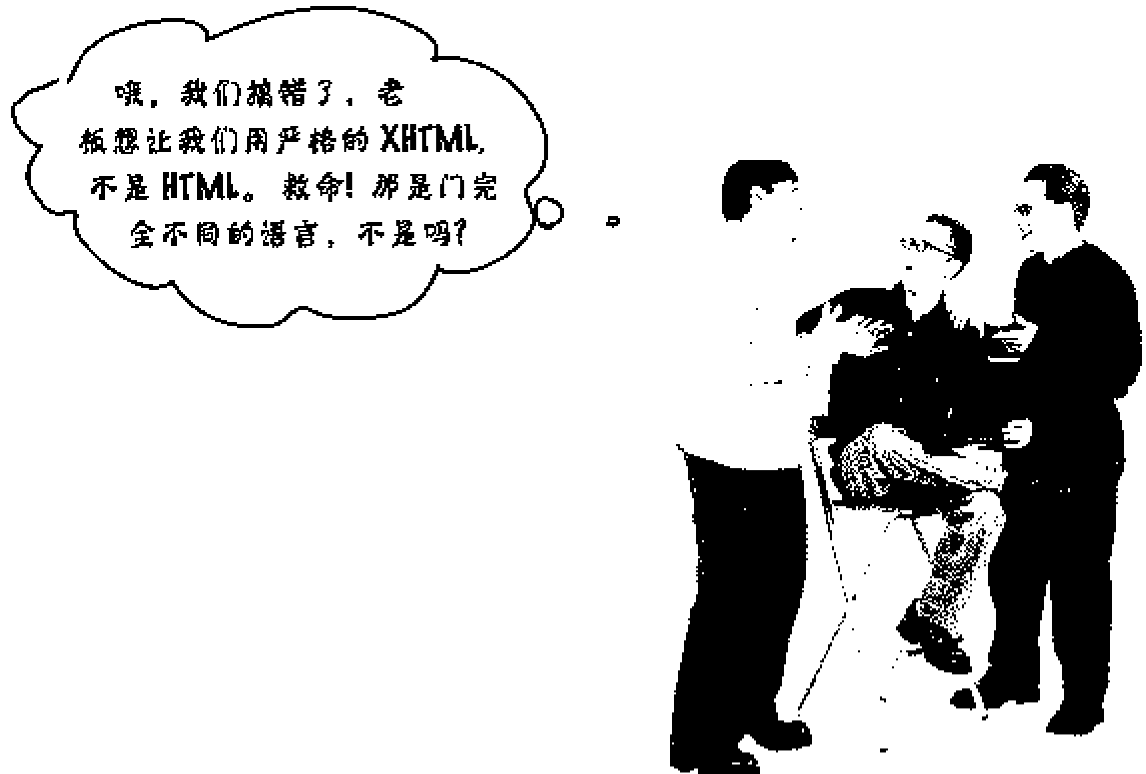


扮演校验器

下面，有一个HTML文件。你的工作是扮演校验器并找出所有的错误。完成后，对照本章末尾的答案，检验你是否已经掌握了全部要点。

做完之后，用校验器检验你的工作。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN"
   "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
   <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
</head>
<body>
   <img src="chamberofcommerce.gif">
   <h1>Tips for Enjoying Your Visit in Webville</h1>
   <p>
      Here are a few tips to help you better enjoy your stay in Webville.
      <ul>
         <li>Always dress in layers and keep an html around your
             head and body.</li>
         <li>Get plenty of rest while you're here, sleep helps all
             those rules sink in.</li>
         <li>Don't miss the work of our local artists right downtown
             in the CSS gallery.</li>
      </ul>
   </p>
   <p>
      Having problems? You can always find answers at
      <a href="http://www.headfirstlabs.com"><em>Head First Labs</em></a>.
      Still got problems? Relax, Webville's a friendly place, just ask someone
      for help. And, as a local used to say:
   </p>
   <blockquote>
      Don't worry. As long as you hit that wire with the connecting hook
      at precisely 88mph the instant the lightning strikes the tower…
      everything will be fine.
   </blockquote>
</body>
</html>
```



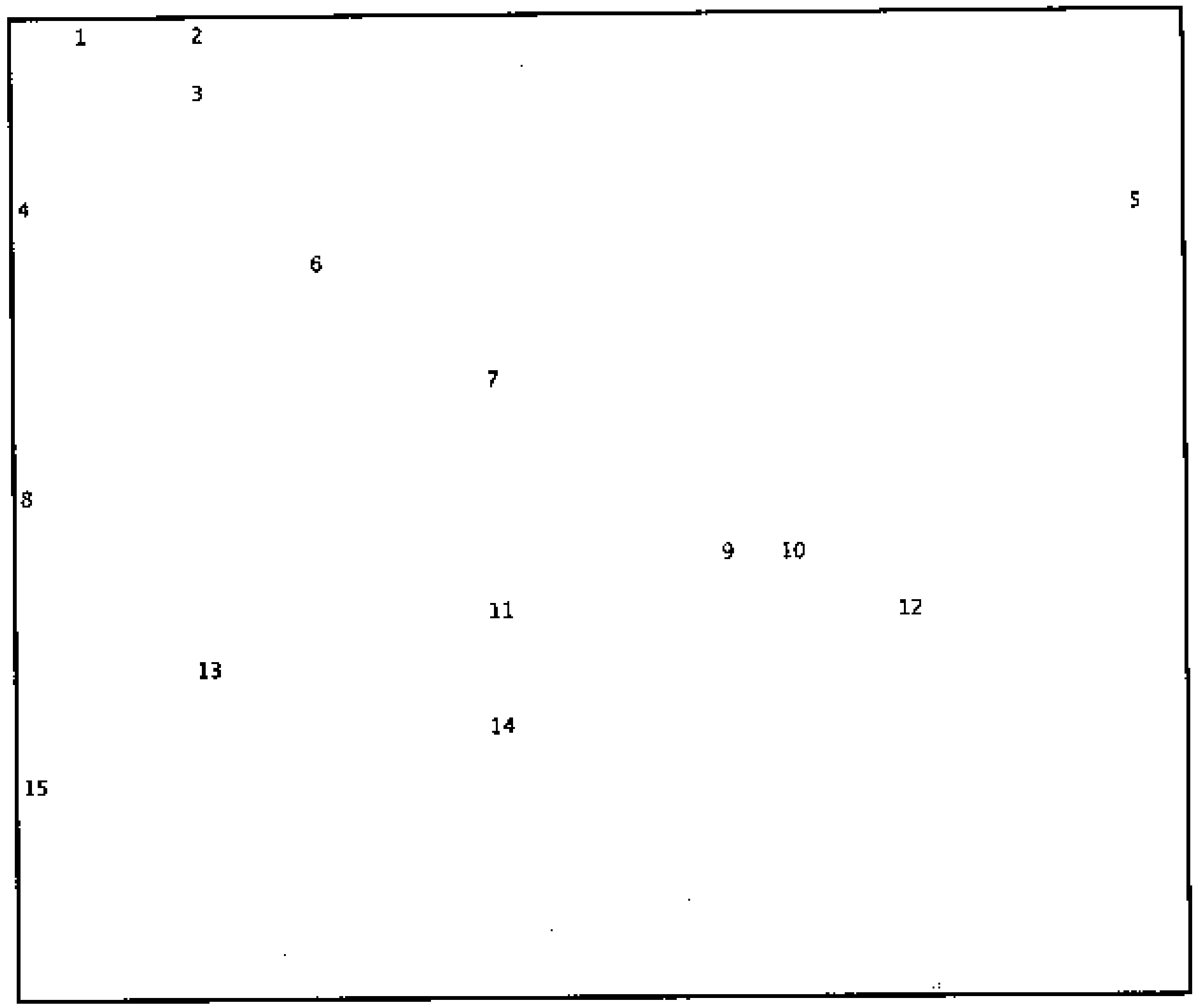
要点

- HTML 4.01是最受浏览器广泛支持的HTML标准。
- 万维网联盟（W3C）是定义“标准HTML”的公认组织。
- 许多浏览器有两种显示HTML的模式：处理旧版HTML的“转换显示”模式和处理HTML 4.01的标准模式。
- 如果你没有告诉浏览器你使用的是哪个HTML版本，许多浏览器将使用“转换显示”模式，而这可能会导致在不同浏览器上有不一致的显示结果。
- 文档类型定义（DOCTYPE）用来告诉浏览器网页是使用哪个版本的HTML。
- 如果你写的是完全合法的HTML 4.01，就使用严格的DOCTYPE。
- 如果你用的是包含面向显示的元素和属性的过渡HTML，那就使用过渡的DOCTYPE。
- <head>元素里面的<meta>标记告诉浏览器关于网页的附加信息。
- 字符编码告诉浏览器网页中使用的字符类型。
- 现今计算机使用的大部分的西欧语言都可以用字符编码ISO-8859-1表示。
- W3C校验器是一种校验网页是否符合标准的免费在线服务。
- 使用校验器检验以确保HTML的结构合理及元素和属性符合标准。
- 遵从标准将加速网页显示，并减少网页在不同的浏览器之间的显示差异。



HTML填字游戏

本章就要结束了。坐下来喝着你喜欢的饮料，集中精力完成下面的游戏。答案都出自本章。



横排提示：

3. 对或错：元素名称应当是小写的。
4. <head>元素中必需的内容。
6. 提供校验器的标准组织。
7. 当你的HTML满足标准时，称作……
8. Microsoft对抗Netscape。
9. 老板要求添加到HTML之前，它是标准的。
11. 标准HTML中<img>属性所必需的。
14. 接受HTML旧标记的DOCTYPE。
15. 用来告诉浏览器和校验器你使用哪种HTML的声明。

竖排提示：

1. 该设备将核查你的HTML是否合法。
2. 在HTML早期，这与HTML结构合在一起。
5. alt属性必需的前提。
6. 浏览器大战的受害者。
7. 编码告诉浏览器你使用的是哪种字符。
10. 要求你的HTML完全遵循HTML 4.01的DCOTYPE 版本。
12. 如果浏览器不知道网页使用的HTML版本，它使用这个模式。
13. 告诉浏览器关于网页信息的标记。

练习答案



扮演校验器

下面，你会发现一个文件。你的工作是扮演校验器并找出所有的错误。这是答案。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN"
   "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
   <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
</head>
<body>
   <img src="chamberofcommerce.gif">
   <h1>Tips for Enjoying Your Visit in Webville</h1>
   <p>
      Here are a few tips to help you better enjoy your stay in Webville.
      <ul>
         <li>Always dress in layers and keep an html around your
             head and body.</li>
         <li>Get plenty of rest while you're here, sleep helps all
             those rules sink in.</li>
         <li>Don't miss the work of our local artists right downtown
             in the CSS gallery.</li>
      </ul>
   </p>
   <p>
      Having problems? You can always find answers at
      <a href="http://www.headfirstlabs.com"><em>Head First Labs</em></a>.
      Still got problems? Relax, Webville's a friendly place, just ask someone
      for help. And, as a local used to say:
   </p>
   <blockquote>
      Don't worry. As long as you hit that wire with the connecting hook
      at precisely 88mph the instant the lightning strikes the tower...
      everything will be fine.
   </blockquote>
</body>
</html>
```

<title>应当在<head>里面。

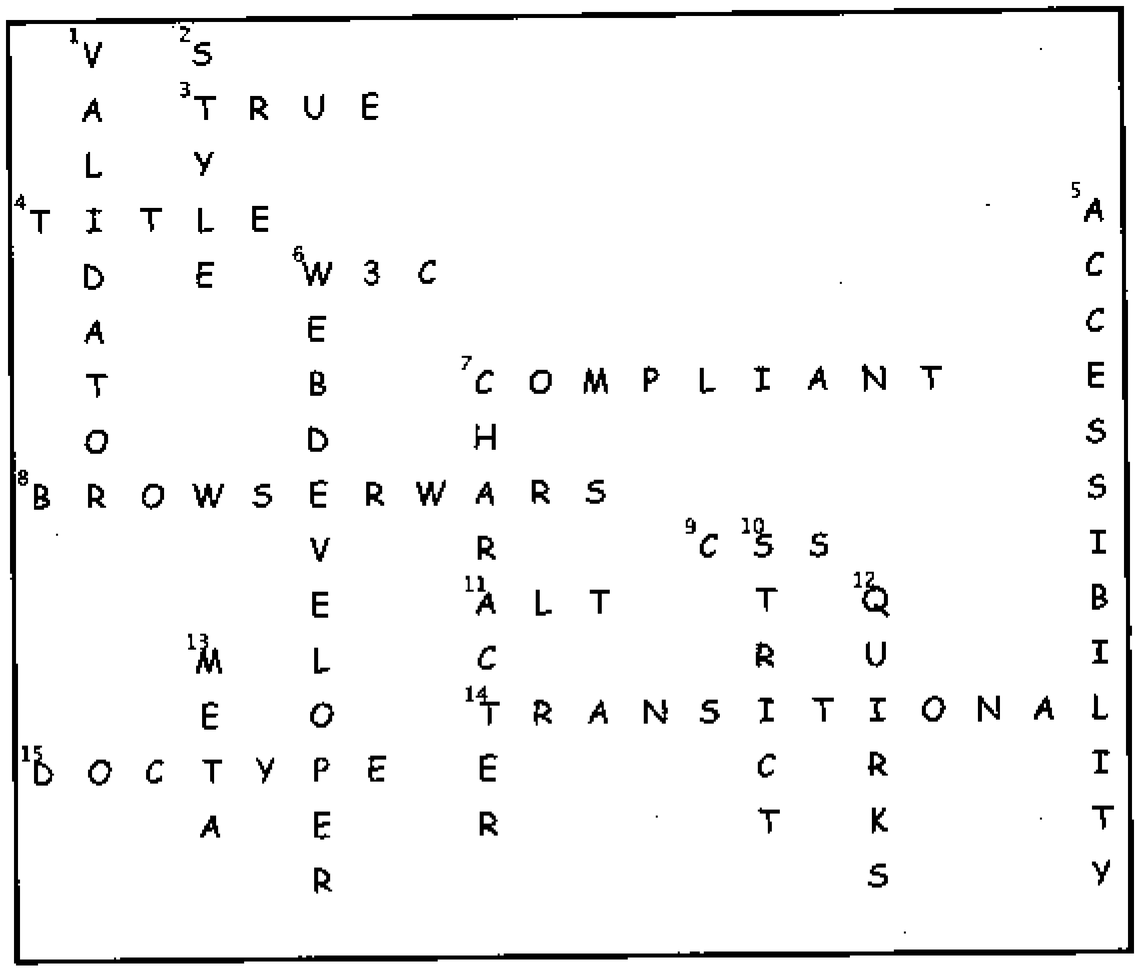
内联元素在<body>的顶行。

没有alt属性。

块元素在<p>里面。

只有块元素才能直接嵌入<blockquote>。

答案



练习答案

轮到你了。把严格的DOCTYPE和<meta>标记添加到“directions.html”和“elixir.html”文件中。试着校验它们是否合法？如果不合法，修改并重试直到它们合法。

答案：为了使“elixir.html”合法，必须给每个<img>元素添加alt属性。

7 转到XHTML

添加一个“X”到HTML



我们一直对你隐瞒着一个秘密。你以为自己买了一本HTML书，但其实这是一本化了妆的XHTML书。事实上，一直以来我们教你的大多是XHTML。你一定很好奇，XHTML究竟是什么？好了，来认识HTML的扩展版本——即所谓的XHTML——HTML的下一版本。它更简单，通俗，并且在一系列的浏览器上更具兼容性。在这简短的一章里，我们将通过三个步骤，由HTML过渡到XHTML。好了，现在翻开书本，我们要开始学习了……（之后我们将学习CSS）。



Joe：我觉得经理自己都不知道XHTML是什么。

Frank：嘿，伙计们，XHTML是HTML的新标准。不会有什么HTML 5.0，新的标准是XHTML 1.0。

Jim：好极了，但我们有必要走在时代前列吗？

Frank：实际上，XHTML 1.0早在2000年就面世了，它也没有想象的那样尖端。

Jim："X"表示什么？听起来挺酷……X-Men,X-Games,X-File,gen-X,而现在是X-HTML？

Frank：很好的想象，但不是那个意思。XHTML里的X表示"eXtensible"，是以XML为基础的另一种说法。

Joe：那些软件开发者不是用它保存数据的吗？

Frank：对。XML表示可扩展的标记语言。

Joe：哦，我看到下面有它与超文本标记语言的对照。

Frank：是的，确实是这样，JOE。XML就如HTML，是一种标记语言，但你几乎可以用它来做任何事，甚至是标记网页。来，让我给你演示一下。

什么是XML?

好了，我们将用一到两页的篇幅回头来介绍一下XML（别与XHTML混淆了）。这不会花很长时间，所以坚持一下……

让我们用HTML与之对比一下。HTML会告诉你哪些元素可以用，哪些元素不可以用。如果想要创造一个元素（比如<cool>），你却办不到。但是如果用XML，你就可以办到了。如果付诸努力，你们还可能用XML开发一套完整的新的标记语言。让我们看一个例子：

这是根元素，它不是被称为<html>，而是<recipe>，因为这是一种XML方法。注意，它有一些HTML中<html>元素没有的额外属性。

```
<recipe xmlns="http://www.foodnetwork.com/recipe" lang="en" xml:lang="en">
   <name>Head First Lounge Iced Tea</name>
   <description>A brisk iced tea with a bit of a kick. We
              serve this all day long.
   </description>
   <ingredients>
      <ingredient measurement="6 cups">water</ingredient>
      <ingredient measurement="2 bags">black tea</ingredient>
      <ingredient measurement= "2 bags">earl grey tea</ingredient>
      <ingredient measurement="6 cups">ice</ingredient>
   </ingredients>
   <preparation>
      <time duration="10 minutes" />
      <step>Boil one cup of water in a pan, remove pan, and
          add tea. Let steep for five minutes.</step>
      <step>Add ice to a pitcher, then add tea,
          then 5 cups cold water.</step>
      <step>Mix well and serve. Give tea a
          quick shake in a shaker for an
          extra touch.</step>
   </preparation>
</recipe>
```

看这些标记，不再有<h1>和<p>了，取而代之的是<recipe>，<name>，<description>，<ingredient>，<preparaton>等，

只需看元素名称，你就能知道这是一个recipe了。

这个空元素看起来有点陌生。我们以后还会细说。

除了元素名称，元素应用的方法类似于HTML。（开始标记，结束标记等）。

BRAIN POWER

回想一下你是如何使用HTML创建一个网页来表示一个recipe的。这与使用XML有何不同?

如何将HTML改为XHTML？

XML是一种可以用来开发新的标记语言的语言，而HTML只是一门标记语言，那么，我们可以用XML改造HTML吗？可以！让我们先看看下面的代码，然后讨论为什么要这么做：

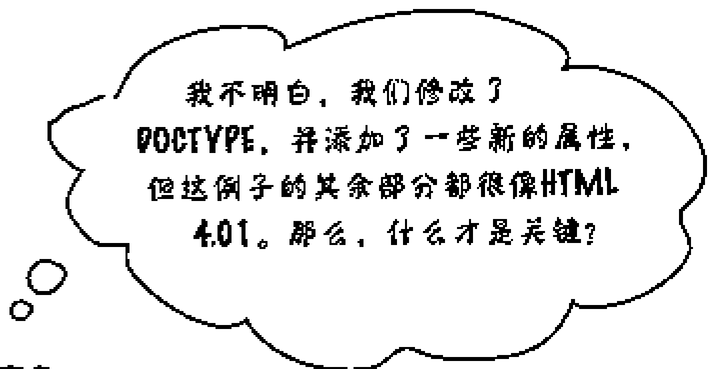
这是DOCTYPE。之前已经学过，但注意了，现在我们使用的是XHTML 1.0而不是HTML 4.01。

和recipe一样，<html>元素现在有xmlns属性、lang属性和xml:lang属性。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
                      "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"lang="en" xml:lang="en">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <title>Head First Lounge</title>
  </head>
  <body>
    <h1>Welcome to the New and Improved Head First Lounge</h1>
    <p><img src="drinks.gif" alt= "The drinks" /></p>
    <p>
      Join us any evening for refreshing
      <a href="elixir.html">elixirs</a>,
      conversation and maybe a game or two
      of <em>Dance Dance Revolution</em>.
      Wireless access is always provided;
      BYOWS (Bring Your Own Web Server).
    </p>
    <h2>Directions</h2>
    <p>
      You'll find us right in the center of downtown
      Webville.  If you need help finding us, check out our
      <a href="directions.html">detailed directions</a>.
      Come join us!
    </p>
  </body>
</html>
```

这里，空元素末尾有一个怪异的"/>"，除此以外，一切如常。

但其余的HTML极像严格的HTML 4.01。嗯，XHTML看起来很像HTML。





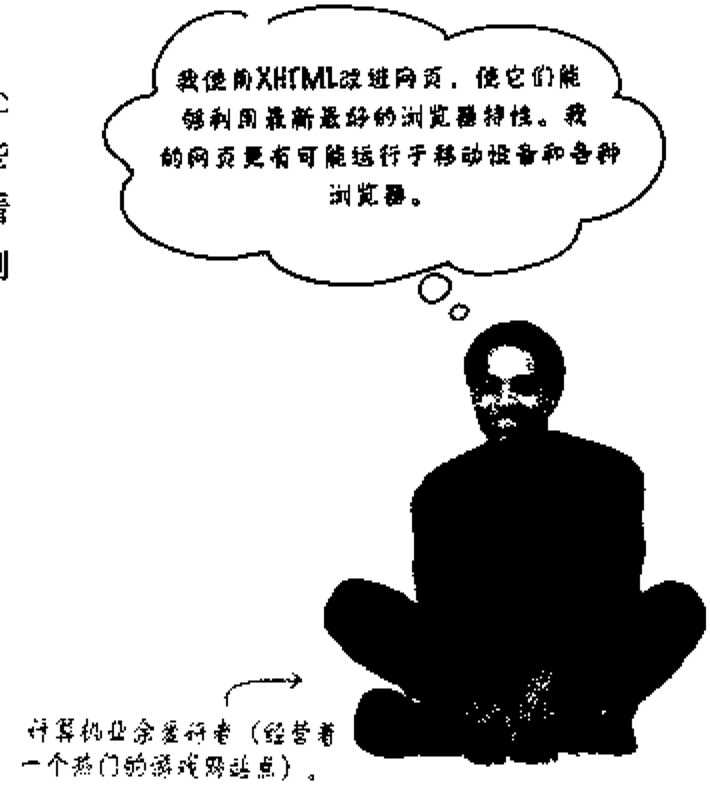
听起来很让人泄气，但是XHTML是XML，而HTML就仅仅是HTML。最大的差别就是，XML的优势在于一旦你用XHTML编写网页，就可以实现许多用HTML不可能实现的功能。这一点在刚接触XML时可能会比较难理解，不过很快就能有所体会。而这也是W3C和其他厂家在拥有HTML的情况下，克服重重困难开发XHTML的原因。

换个角度看待这问题：鉴于以前的网页，我们可以发现XHTML和HTML的不同之处。你使用新的DOCTYPE，并对属性和空元素的使用方法做了一些小的修改。这些小修改就是把HTML转化为XHTML的全部要求。

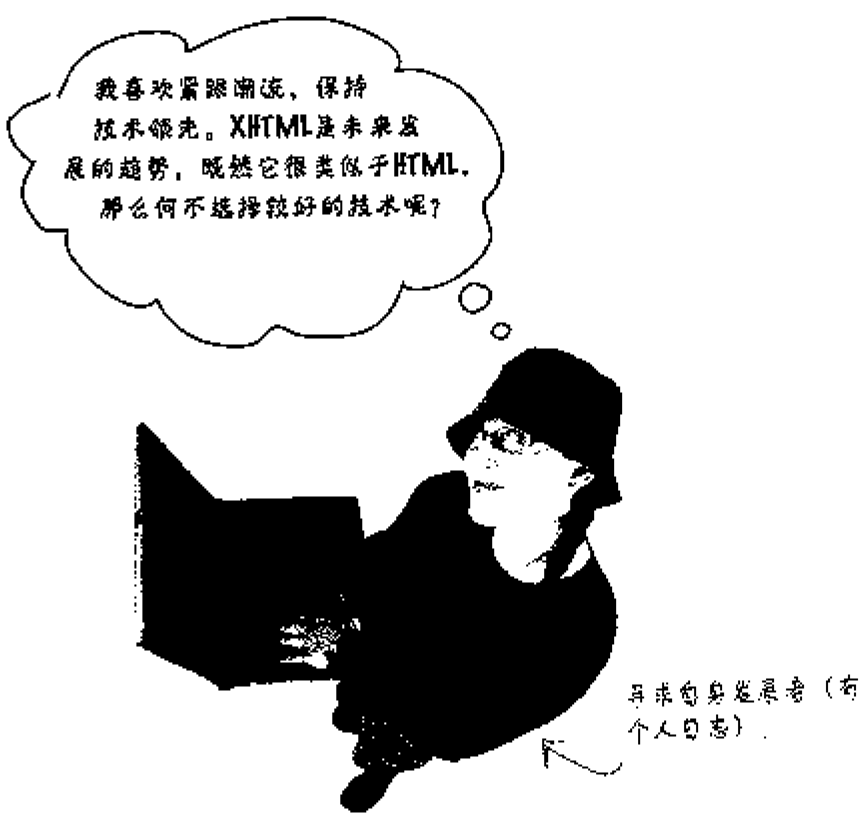
既然学会XHML是如此容易，那它又有什么好处呢？

为什么要使用XHTML？

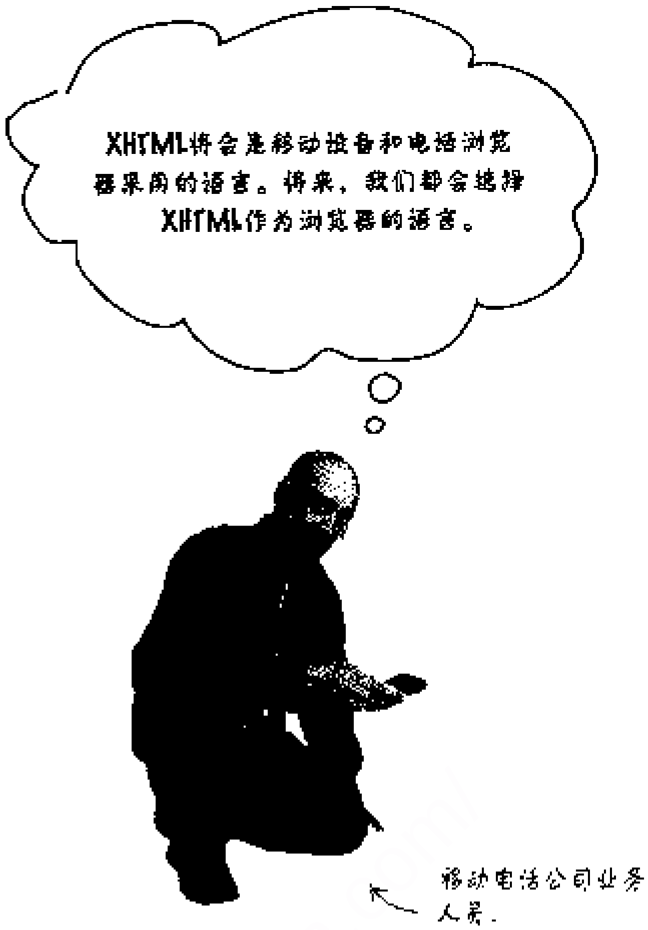
通过使用严格的HTML 4.01，你已经体会到XHTML的一些好处了。然而，因为XHTML是XML，它相对于HTML 4.01更有一些优势。通过使用过XHTML的开发人员的反映，让我们一起看看XHTML带来的好处，包括你使用严格的HTML 4.01时所体会到的好处。

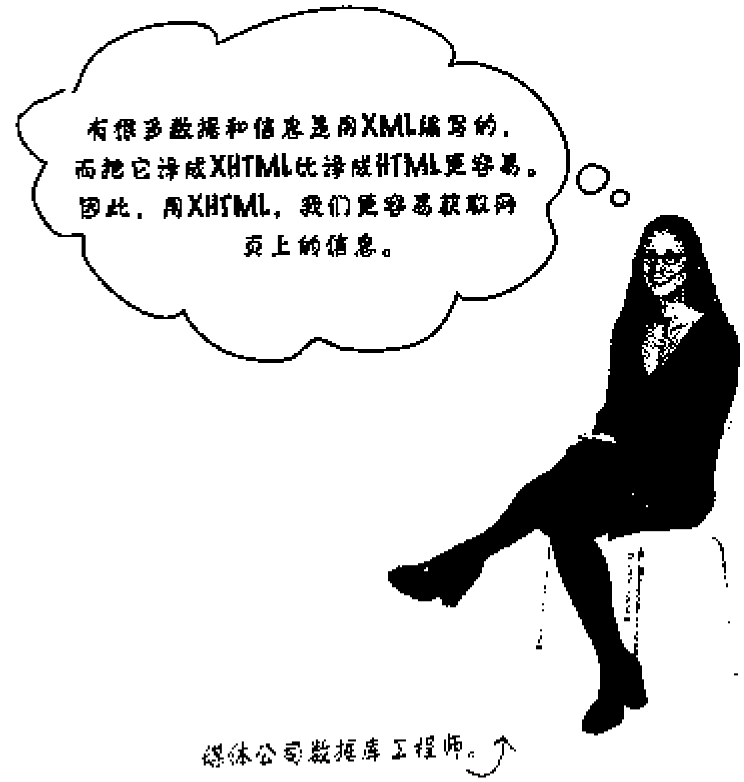









XHTML将会是移动设备和电话浏览
器采用的语言。将来，我们都会选择
XHTML作为浏览器的语言。
移动电话公司业务
人员。


有很多数据和信息是用XML编写的，
而把它转成XHTML比转成HTML更容易。
因此，用XHTML，我们更容易获取网
页上的信息。
媒体公司数据库工程师。


能够读取XML的现行软件应用
程序就能够读取XHTML。
高级软件开发人员。


XHTML兼具XML和HTML的优点。既能
像XML那样存储大量结构化的文档，也
能像HTML那样可以用CSS设计外观。
大城市图书馆的图
书管理员。

你几乎已经在使用XHTML了，比你自己想象的还要接近

即使HTML和XHTML十分相似，但它们之间还有一些细微的差别。以下是关于HTML 4.01转化为 XHTML 1.0的简单清单：

XHTML 1.0清单

要把HTML改为XHTML，需要做以下几件事：

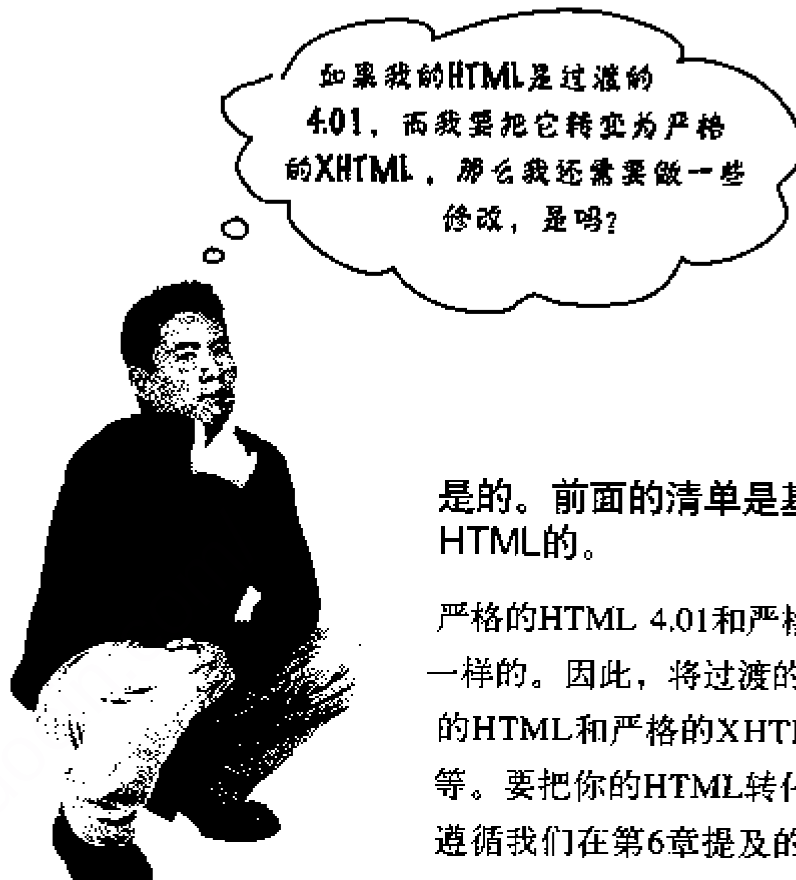
- ☐ 把DOCTYPE更改为严格的XHTML，如果你还在使用过渡的HTML，那你可以把它转化为过渡的XHTML 。
- ☐ 添加`xmlns`属性、`lang`属性和`xml:lang`属性到`<html>`开始标记。
- ☑ `<html>`标记必须是DOCTYPE之后的第一个标记，而`</html>`必须是文档的最后一个标记。
- ☑ 所有的元素名称必须用小写字母表示。
- ☐ 所有的开始标记都须对应有结束标记。如果一个元素是空的，那么它的标记必须以空格结尾，后跟/>。
- ☑ 所有的属性值必须非空，并且带双引号。
- ☑ 在HTML里不能使用&，&表示实体的开始，因此，应当用`&`；此外，把特殊字符转化为实体。

我们已在你做过的要求前打上勾。所以想要转化为XHTML 1.0，要做的已经很少了。

我们将讨论一下这是什么意思。

如果你从本书开始学习HTML，并一直使用严格的HTML 4.01，那么转向XHTML 1.0对你来说将是一个简单的过程。你只须注意一些必要的东西，而后面我们将会就此进行讨论。

相反，如果你有许多遗留下来的HTML需要修改，那么可能会有很多工作要做。但是，你可以借助一些工具。我们也将就此进行讨论。



是的。前面的清单是基于你正在写严格的HTML的。

严格的HTML 4.01和严格的XHTML 1.0基本上是一样的。因此，将过渡的HTML 4.01转化为严格的HTML和严格的XHTML所需的工作量大约相等。要把你的HTML转化为其中一个，首先必须遵循我们在第6章提及的所有要求，删除外观标记并整理你的HTML。

过渡的XHTML 1.0版本本质上与过渡的HTML 4.01一样。它们都认可被严格版本排斥的外观元素，并且内联元素可以直接嵌入网页的body部分。如果你想使用它，记得使用过渡XHTML1.0的DOCTYPE，而不是严格的DOCTYPE。

通过三个步骤，将严格的HTML转化为XHTML 1.0

1 将DOCTYPE改为严格的XHTML 1.0。

你已经了解了DOCTYPE，并且对严格的HTML 4.01文本类型也已经习以为常。同样有一个文本类型代表严格的XHTML 1.0，所以必须对DOCTYPE做些改动。形式如下：

就如HTML DOCTYPE，这是一个公共文档类型。

这表示使用XHTML 1.0——严格的XHTML版本。

```
<!DOCTYPE html
     PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

这有一个URL指向严格的XHTML 1.0的定义。

2 添加xmlns属性、lang属性和xml:lang属性到<html>元素。

记着，除了XHTML，XML能被用来定义许多种标记语言。为了保证所有这些标记语言是直观明了的，当你使用**<html>**元素时，XML需要知道你正在使用哪种标记语言（毕竟一些人会使用自定义的XML语言，譬如称之为“Hippo Tipping Markup Language”，如果不加以说明会造成混淆）。所以为了使事情清晰明了，**xmlns**属性需要指定**<html>**元素属于哪种语言。那么**<html>**元素中剩余的那些元素呢？默认情况下它们将继承父类的**xmlns**属性。

<html>元素也需要**lang**和**xml:lang**属性,用来指定在XML文档中使用的语言。下面是在XHTML中**<html>**起始标记的用法：

```
<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">
```

xmlns属性用来指定“html”属于哪一种XML语言。

XML用URL作为唯一的标识符来表示一种语言。如果有人用“Hippo Tipping Markup Language”编写网页，那么他们可能用“http://www.hippotipping.com/tml”作为标识符。URL的内容无关紧要——单独的URL足够保证它是唯一的。

必须指定我们使用的是英语。由于XHTML的路径是由浏览器解释的，你可能只使用其中的一种，最好是两种都用。

❸ 所有的空标记都应以“/>”结尾，而不是“>”。

这是从HTML转化为XHTML 1.0的最后步骤，也是最古怪的步骤。但是如果你知道诀窍，那这也就没什么神秘了。我们已经知道XHTML比HTML更严谨，而它之所以更严谨，原因之一就是因为它有结束标记。在HTML中，可以有一个没有结束标记的空元素。但是在XHTML里，如果没有结束标记，就必须在最后的“>”前加一个斜杠来告诉浏览器你要结束这个元素。以**
元素为例，在HTML里，我们只须写
；但是在XHTML里，要写成
。末尾的斜杠告诉浏览器无须等待结束标记了，因为那里只有
**。现在，你可能已经注意到在“/>”前没有空格。那是因为XHTML对此没有要求。然而，如果斜杠前没有空格，一些较旧的浏览器就无法识别“/>”，因此，为了向后兼容，就在“/>”里的斜杠前加一个空格吧。

看看下面两个例子，你就知道该如何把HTML空元素转化为XHTML空元素了：

旧版的HTML 4.01

```
<br>

<img src="drinks.gif" alt="Drinks">
```

没有结束标记？在HTML里没问题。

新的改进了的 XHTML 1.0

```
<br />

<img src="drinks.gif" alt="Drinks" />
```

但在XHTML里，我们必须声明我们的意图。如果元素是空的，在最后的“>”前加“/”让浏览器知道你的意图。

在斜杠之前插入一个空格以给那些较旧的浏览器一个提示。

练习

看看Tony的网页（在第3章，记得吗？）并把它改为XHTML。我们已经整理了他的代码并把它们转化为了严格的HTML 4.01——我们把他的<img>元素嵌入<p>元素中，添加了alt属性，把柏玛刮胡膏的标语放在一个<p>元素里面，还添加了一个<meta>标记。你将会在“chapter7/journal”文件夹里找到“journal.html”的HTML 4.01严格版。以下是你要做的事：

1. 将DOCTYPE由严格的HTML 4.01改为严格的XHTML 1.0。
2. 添加xmlns属性、lang属性和xml:lang属性到<html>开始标记。
3. 把空元素结尾的“>”改为“/>”。
4. 保存，将网页重新载入浏览器。

本章结束时，记得检查你的工作。

there are no Dumb Questions

问：你能再详细解释一下xmlns属性吗？我感觉我还没有完全掌握。

答： 不仅仅是你一人有这个问题。这是XML最不易理解的部分之一。它的工作原理是这样的：许多人可以创建XML语言（就个人来说，我们赞成到外面多看看，但有些人沉迷于这些）。假设两人以相同的名称称呼元素。以名称<table>为例，有些人认为这是HTML里的一个元素；其他人却认为这是XML的一部分。所以，如果你在XML里使用<table>，我们如何知道你指的是哪一个呢？这就是引进xmlns属性的原因。xmlns属性持有一个唯一的标识符，可以用来确定你指的是哪一种语言。如果使用XHTML，那标识符是`http://www.w3.org/1999/xhtml`。

问：但那是URL，不是标识符。

答： 是的，XML使用者都对此很不理解。它看起来就像一个URL，但却被用作标识符。你访问那URL就会发现一些关于语言的情况。但这些东西不是必需的。

问：这真的是XHTML吗？为什么根元素是<html>，而不是<xhtml>？

答： 因为XHTML要与HTML兼容。如果把根元素改为<xhtml>，那么那些旧的浏览器就不知道如何显示网页了。

问：先前你提到一些可以帮助我把HTML转化为XHTML的工具。

答： 是的。有一个小工具叫做Tidy，它能帮你规范文本并为转化成XHTML做好准备。Tidy有很多功能，并且能够完成很多使不合格的HTML变为合法的重要任务。它还可以删除许多遗留的影响外观的HTML，而用CSS代替。你可以在http://tidy.sourceforge.net找到Tidy。

问：那么，如果我已经有了严格的HTML，那我该做的就是把它转化为XHTML吗？

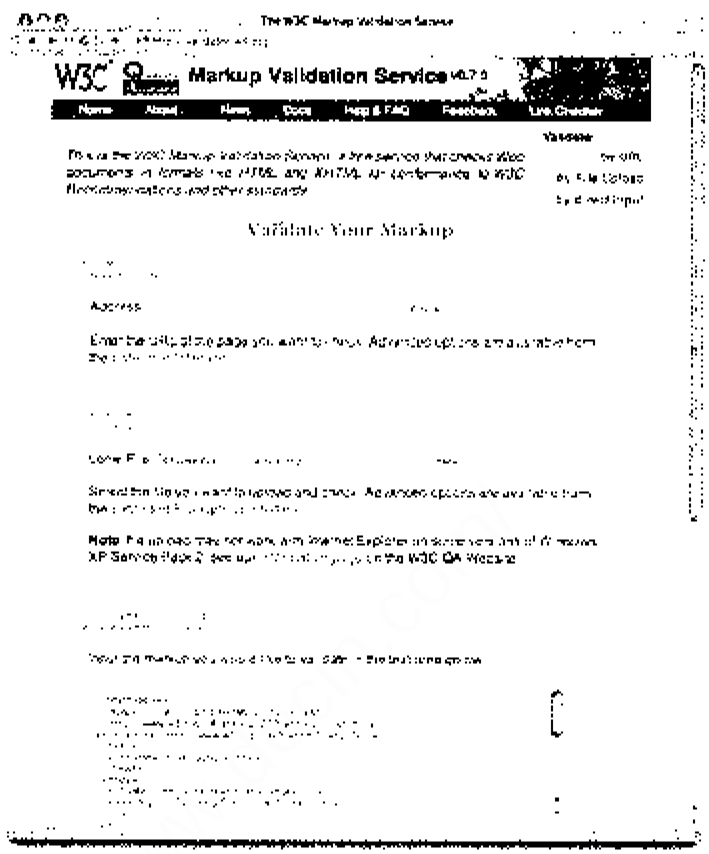
答： 没错。让我们试一试吧……

校验：不只适用于HTML

第6章之后，你已经是一个使用W3C校验器的专家了，并且你会发现校验器是最新的，并为校验你的XHTML做好了准备。

你可以用和校验HTML一样的方法来校验XHTML。

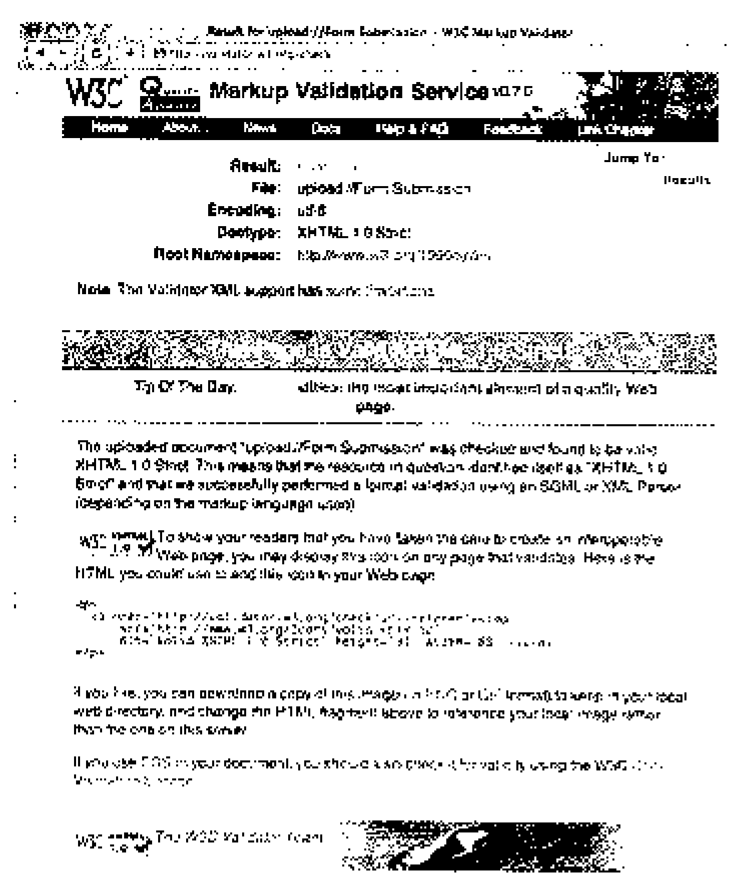




打开validator.w3.org并且附上你的XHTML，或者上传它，或者向校验器提交你的URL。

校验器将核查你的XHTML，并向你报告它是合法的还是包含错误的。

校验器



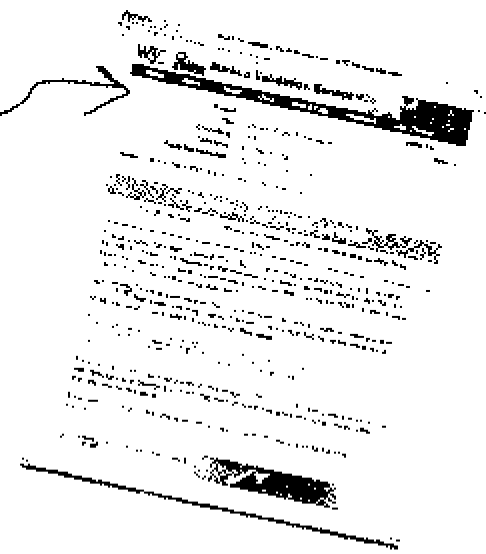
there are no Dumb Questions

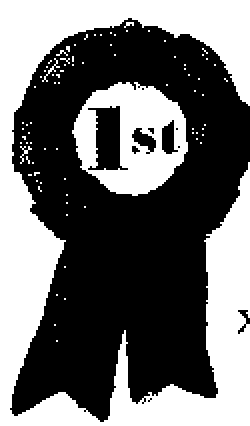
问： 校验器如何知道我校验的是XHTML还是HTML？我校验HTML也是用这个校验器啊。

答： 校验器读取你的DOCTYPE声明，而声明规定文本是XHTML或过渡的XHTML，校验器就以此为根据确定它校验的是哪一种文本。

练习

如果没有通过校验，你觉得我们会放过你吗？在W3C上校验“chapter7/lounge”文件夹里的“lounge.html”文件和“chapter7/journal/”文件夹里的“journal.html”文件（几页前被你改为XHTML的文件）。如果有错误，就检查打字输入，改正它们，然后重试。

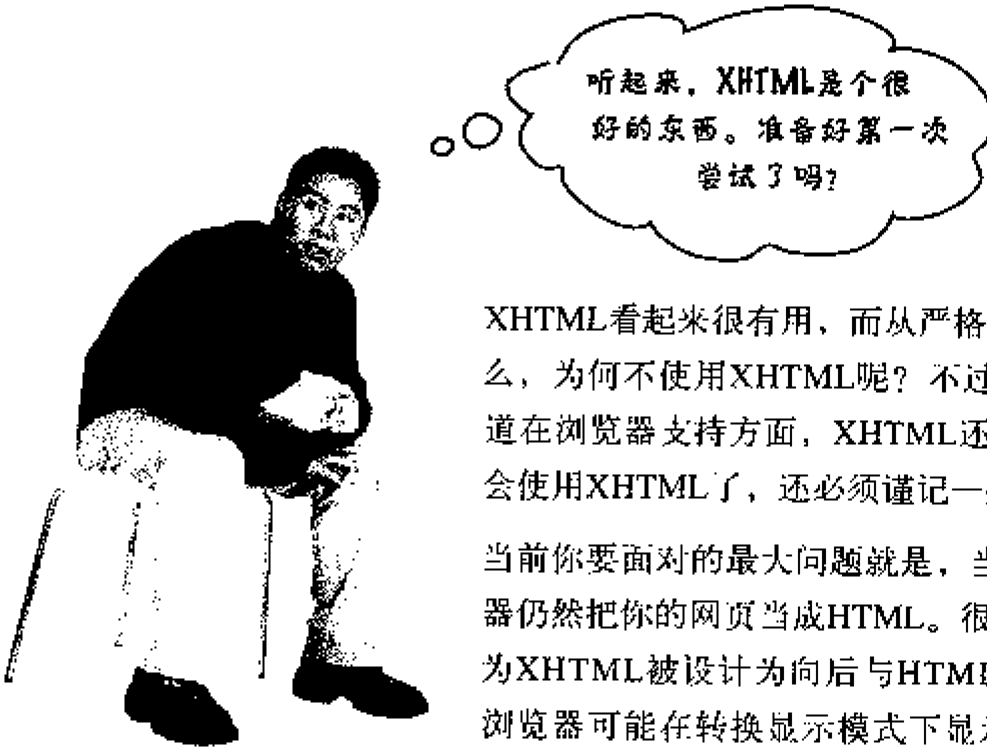




祝贺你，你已经编写了你的第一个XHTML！

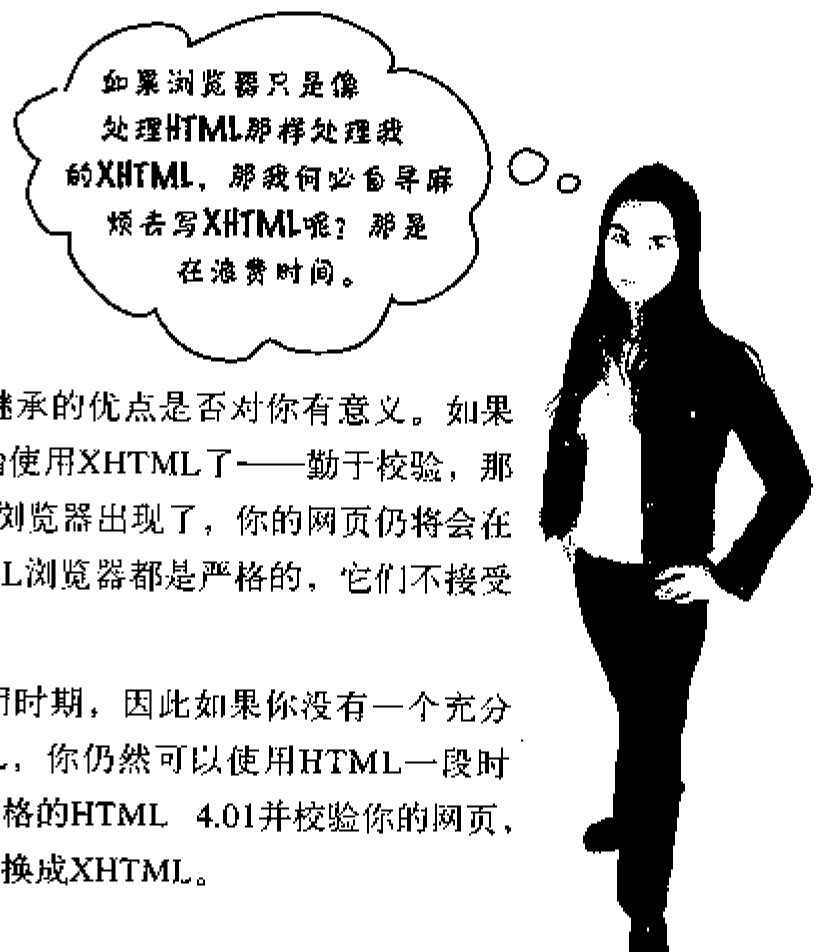
成功了：你已经把你的HTML转化为XHTML了。尽管标记跟以前没有太大的差别，但是接下来XHTML文本将有一系列新的发展前景。不仅如此，你已经能够运用与已学到的技术相似的新技术了。现在你可以去告诉所有的朋友你已经在使用XHTML了（当然，你可以不告诉他们HTML与XHTML没有太大的不同）。

哦，还有一点，XHTML与CSS兼容，你很快就可以将你的第一个设计添加到XHTML网页中了。



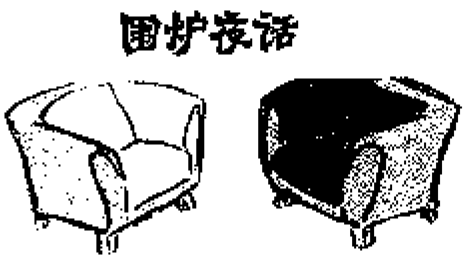
XHTML看起来很有用，而从严格的HTML 4.01转化并不困难，那么，为何不使用XHTML呢？不过在使用XHTML之前，你应该知道在浏览器支持方面，XHTML还是比较超前的。即使现在你已经会使用XHTML了，还必须谨记一些问题。

当前你要面对的最大问题就是，当你使用XHTML时，有一些浏览器仍然把你的网页当成HTML。很多情况下，这没什么大问题，因为XHTML被设计为向后与HTML兼容。然而，在最坏的情况下，浏览器可能在转换显示模式下显示网页（如果你已经忘记转换显示模式，请参看第6章）。因此，你的XHTML可能得到不一致的显示结果。怎么办呢？现在你最好在不同的浏览器上校验XHTML，以确保一切按计划进行。



该弄清XHTML从XML继承的优点是否对你有意义。如果有，那今天你就可以开始使用XHTML了——勤于校验，那么将来当严格的XHTML浏览器出现了，你的网页仍将会在上面很好地工作（XHTML浏览器都是严格的，它们不接受不合法的XHTML）。

HTML还将有较长的使用时期，因此如果你没有一个充分的理由去转换为XHTML，你仍然可以使用HTML一段时间。并且，如果你使用严格的HTML 4.01并校验你的网页，你完全可以在转眼间就转换成XHTML。



今夜话题：HTML和XHTML争赢支持率

HTML

很高兴有机会说服你继续使用我：HTML 4.01。我在很长的一段时间内还是最通用的，无需急于转向XHTML。

XHTML

HTML，面对现实吧，你已经过时了。人们的标准已经改变。我是未来的大势所趋，任何有进取之心的人都会转向XHTML的。

HTML

你我之间大同小异。我是说，4.01和XHTML 1.0基本上是一样的。

XHTML

你怎么会说我们是一样的呢？你是HTML，而我是XML。

HTML

迄今为止，你并没有得到优待。

XHTML

哼，等着吧。每天都有许多支持XHTML的设备增加。很多使用XHTML的应用程序正在递增。

HTML

问题就在这儿：你认为人们的应用程序都使用XHTML，或者每个人都在为移动设备做网页。有些人却只是希望做出好的站点。为什么你非得要求他们遭受这些麻烦呢?

XHTML

其实并不麻烦。如果你已经在使用HTML 4.01，那么只需一小步就能变成XHTML。你所要做的只不过是更改DOCTYPE，添加一些属性到`<html>`元素。还会有什么大问题呢?只需要几分钟的工作，为什么不用最新最好的技术呢?

HTML

XHTML

你忘了一些细节。许多浏览器不能很好地处理XHTML。事实上，它们只是把它当成HTML。因此你做了那么多工作，觉得你的XHTML与众不同，其实只是在骗自己。

嘿，这是好事。XHTML的设计者知道并非所有的浏览器都支持XHTML。所以他们使它向后兼容，也就是说，你可以把网页转化为XHTML，也可以让它在旧的浏览器上运行。

但那有什么价值呢？如果你的XHTML被浏览器当成HTML，那它充其量也不过是HTML罢了！

嗯，但情况正在改变；每天都有越来越多的支持XHTML的浏览器面世。因此，与时俱进吧！这很容易，并且当新的浏览器和设备出现的时候，你就将万事俱备而无需再改变了。

那倒是不错，但我还是要奉劝人们无需太在意。我已经足够好了，完全可以满足他们的需要。许多人没有必要使用XML。

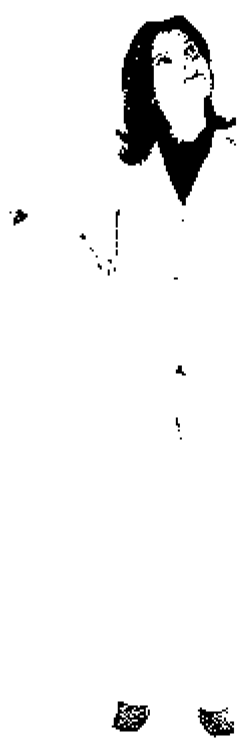
你不能奢望将来XHTML运用于所有领域。XHTML是趋势，如果你现在转向XHTML，那你将不致落伍。

好的，我承认你是对的，XHTML是未来发展的趋势。但你也说了，转到XHTML只是一小步，那么我的用户可以等到XHTML通用时，再迈出那一小步啊。

那句话怎么说来着？“你不能教唆一条老狗耍新花样。”

你的意思是说“你可以赶马下水吗……”

HTML还是XHTML？决定权在于你……



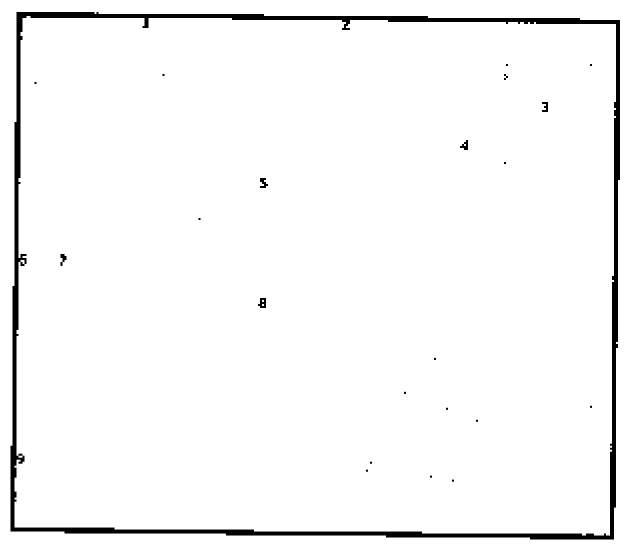
XHTML的优点对你有意义吗？你是否为了Web把现行的XML译成HTML？你是否在做一些必须在移动设备上很好运行的网页，一些最新的网页是否在近期内对你很重要？你想保持技术领先吗？ 如果回答是肯定的，我们有个好消息：现在你就可以转向XHTML。而所有你要付出的只是一个新的DOCTYPE和对一些标记的小改动。现在，不是所有浏览器都支持XHTML，但这是迟早的事，况且，即使它们还不支持XHTML，你的网页依然可以正常显示，因为浏览器将会把它们当成HTML处理（尽管如此，但别忘了我们已经提到的告诫）。那么，好好努力，用心学习XHTML吧！

那些新功能对你来说都不重要？你主要考虑的是制作好的网页？我们又有好消息告诉你：坚持使用严格的HTML 4.01很容易，并且使用浏览器目前选择的语言，这样会比较方便。如果你觉得已经有必要升级到XHTML，你可以按照本章的三步纲要实现升级。

所以，无论你的选择是什么，都有自己的理由。我们希望你是最好的。XHTML与HTML之间的差异真的很细微，那么为何不向前发展转向XHTML呢？我们已使用过XHTML，并且本书剩余的部分也将使用XHTML 1.0。如果由于某些原因，你必须继续使用HTML 4.01，那也没关系。事实上，既然它们基本相同，那对于本书剩余的部分你也不会有什么困难。只须确保为你使用的版本配上正确的DOCTYPE就可以了。

小型的XHTML填字游戏

这一章很简短（你很庆幸吧）。下面是一个小型的XHTML填字游戏。



横排提示：

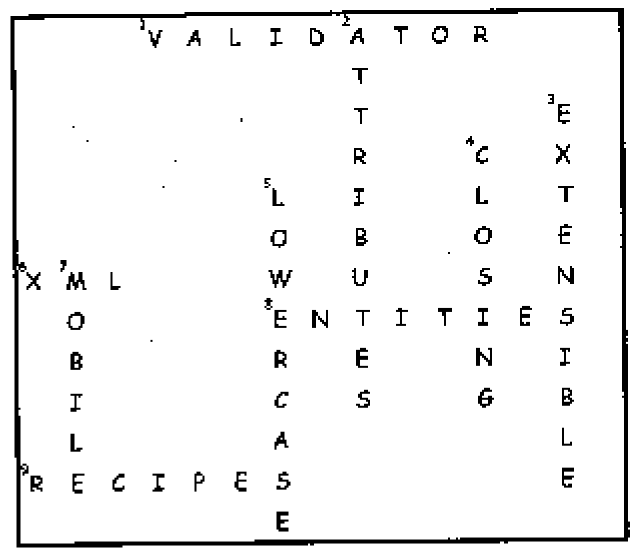
1. 被用来校验你的XHTML。
6. XHTML属于这类标记。
8. 用这些表示特殊字符。
9. 我们为这些发明了一种XML语言。

竖排提示：

2. XHTML要求在<html>元素里附加________。
3. XHTML里的X代表________。
4. 在XHTML里，你必须有________标记。
5. 在XHTML里，所有的元素名称必须是这样的。
7. 这类设备都采用XHTML。

小型的XHTML填字游戏的答案

这一章很简短（你很庆幸吧！）。下面是一个小型的XHTML填字游戏。



练习答案

看看Tony的网页（在第3章，记得吗？）并把它改为XHTML。我们已经将大部分的journal版本放入“hapter/journal”文件夹，你可以在里面找到一个HTML 4.01的“journal.html”严格版本。

这是答案：

```
<!DOCTYPE html
    PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">
  <head>
  <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
    <title>My Trip Around the USA on a Segway</title>
  .
  .
  .
  </body>
</html>
```

将DOCTYPE改为严格的XHTML 1.0。

添加3个属性到<html>开始标记。

<meta>标记也需要一个“/>”。

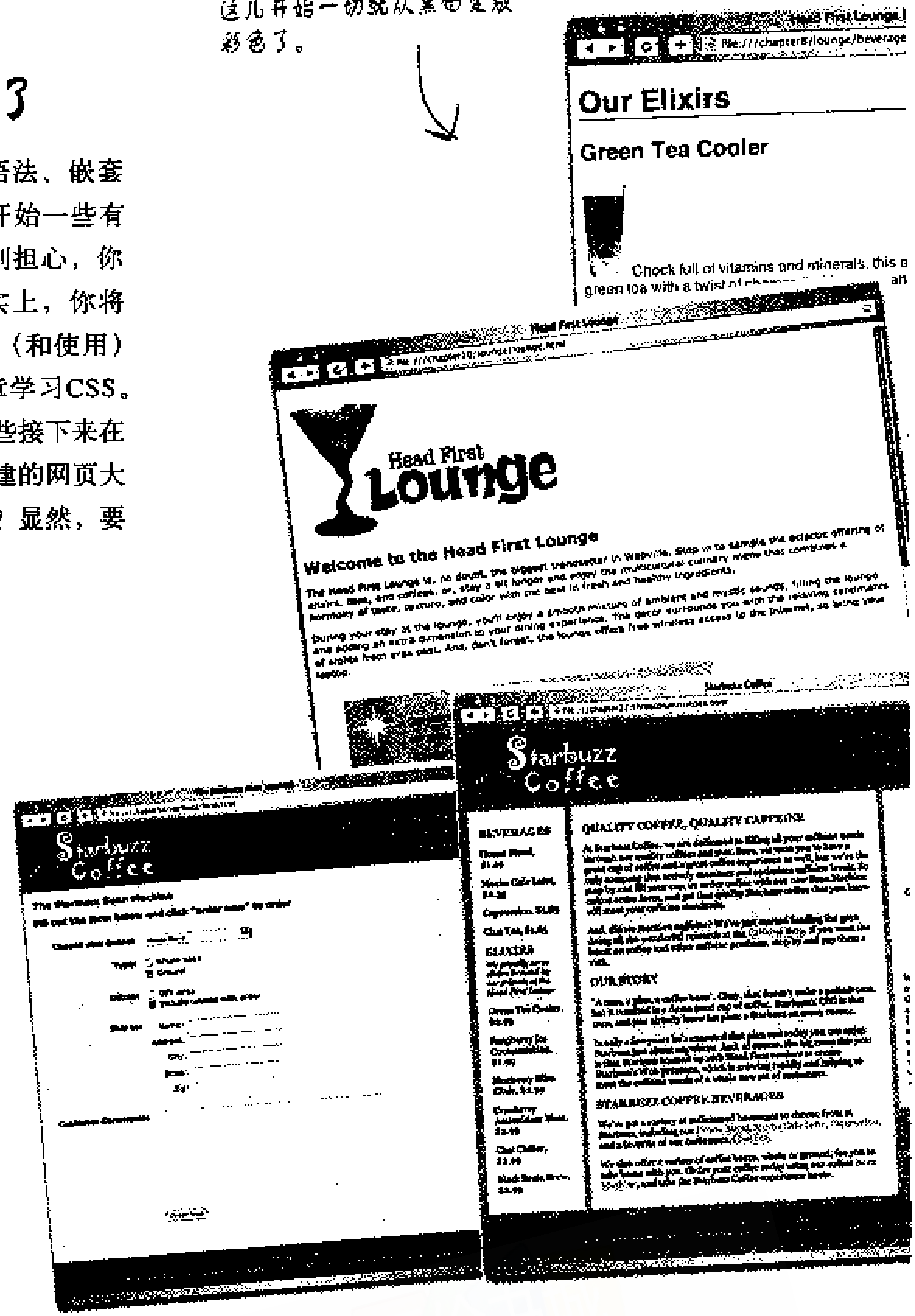
别忘了在空元素处放置“/>”。

添加一些样式



听说这本书将介绍CSS。 迄今为止，你一直专注于学习用XHTML创建网页结构。但是如你所见，浏览器对网页的样式有很高的要求。当然，你可以借助一些工具，但那是不必要的。使用CSS，你可以完全控制网页的外观，甚至不用更改XHTML。它真的可以这么容易就能实现吗？好了，你将要再学一门新的语言，毕竟，在Web镇上可以同时使用两种语言。读完本章对CSS语言学习的介绍，相信你会对网页设计有更全面的了解。

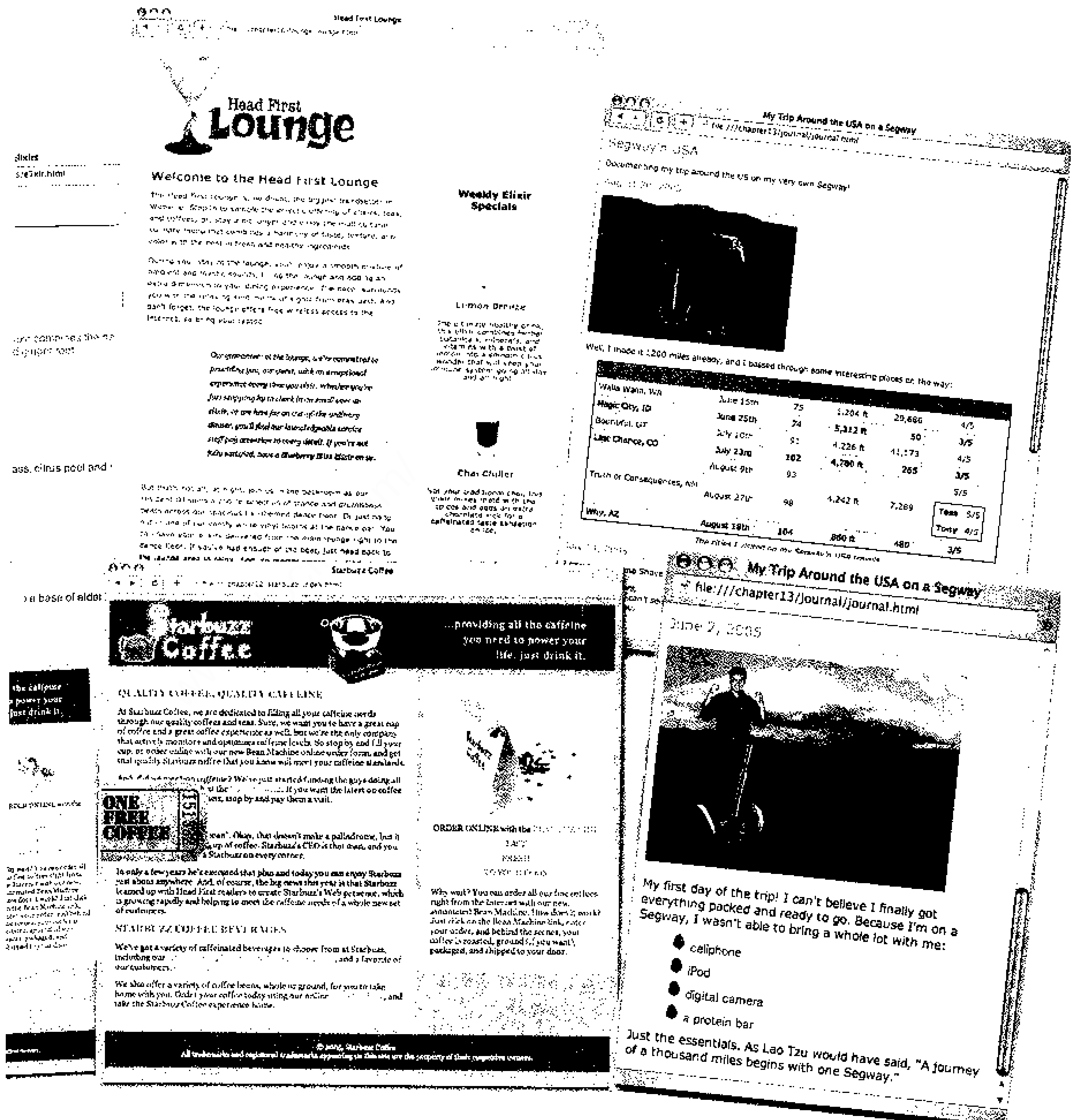
还记得 Wizard of Oz吗？从这儿开始一切就从黑白变成彩色了。



你不再处于勘萨斯州了

你已经就标记、结构、校验、基本的语法、嵌套和规范进行了动态的学习，现在你要开始一些有趣的工作了，那就是修饰你的网页。别担心，你所学的XHTML知识不会被浪费。事实上，你将会发现透彻理解XHTML对CSS的学习（和使用）是至关重要的。我们将在接下来的几章学习CSS。为了给你点启发，这两页我们展示了一些接下来在书中你将会接触到的设计。与你之前创建的网页大为不同。那么，你应该怎么创建它们呢？显然，要学习CSS。

让我们开始吧……


Head First Lounge
Welcome to the Head First Lounge
Weekly Elixir Specials
Lemon Breeze
Chai Chiller
My Trip Around the USA on a Segway
Segway'n USA
Documenting my trip around the US on my very own Segway!
Well, I made it 1200 miles already, and I passed through some interesting places on the way:
Starbuzz Coffee
...providing all the caffeine you need to power your life. just drink it.
QUALITY COFFEE, QUALITY CAFFEINE
ONE FREE COFFEE
My Trip Around the USA on a Segway
file:///chapter13/journal/journal.html
June 2, 2005
My first day of the trip! I can't believe I finally got everything packed and ready to go. Because I'm on a Segway, I wasn't able to bring a whole lot with me:
cellphone
iPod
digital camera
a protein bar
Just the essentials. As Lao Tzu would have said, "A journey of a thousand miles begins with one Segway."

Web镇"贸易区"之见闻

没留意最新的新闻实播？没关系，这里简要介绍一下：有两个邻居，两幢房子和1000美元。这两个邻居相互交换房子，为对方进行装修，并且要使用这1000美元，要在48小时内，彻底重新设计一或两个房间。让我们考虑一下……



当然，在Web镇的展览上，每个人谈论的都是CSS设计。如果你不太了解它们，这儿有一些提示：CSS里面的每个语句都包括一个场所（就像卧室），一个属性（就像窗帘或地毯）和一个提供给属性的样式（就如蓝色，或者一寸的瓷砖）。

使用CSS设计XHTML

我们都确信CSS在房屋设计领域有一个光明的前景，但让我们先回到XHTML。XHTML没有房间，但它有元素并且那些元素就是我们将要设计的场所。想要把你的<p>元素的墙漆弄成红色吗？没问题；只不过段落没有墙，你要设置的是段落的**background-color**属性。如下：

你要做的第一件事是选择你要添加样式的元素，比如<p>元素。注意，在CSS中，元素名称的两边不加<>。

提着你应指定你所添加样式的属性，比如说<p>元素的背景颜色。

你要将背景颜色设置为红色。

```
p {
        background-color: red;
}
```

最后，加一个分号。

将<p>元素的所有样式置于大括号内。

在属性和它的值之间有一个冒号。

我们称整个过程为规则。

你也可以将它写成：

```
p { background-color: red; }
```

我们做的就是除去换行。就如XHTML，你可以随心所欲地安排CSS的格式。对于较长的规则，你经常要添加换行或缩进以使CSS更易阅读。

想要添加更多样式？

你可以在每个CSS 规则里添加若干个属性和值。如果你也希望给你的段落加一个边框，像这样：

```
p {
        background-color: red;
        border: 1px solid gray;
}
```

你要做的就是再添加一个属性并给它赋值。

<p>元素将有一个边框……

……宽度为一个像素，实线，灰色。

there are no Dumb Questions

问： **每个<p>元素都拥有相同的样式吗？或者说，我可以把两个段落设置成不同的颜色吗？**

答： 迄今为止，我们使用CSS规则来为所有的段落定义样式，但CSS是富有表现力的：它可以用许多不同方法指定样式，来设计不同的元素——甚至元素的部件。稍后介绍如何将段落设置为两种不同的颜色。

问： **可以在一个元素上设置什么属性呢？**

答： 有许多属性可以被设置到元素里，多到记不下来。接下来几章你将熟悉更多常见的属性。或许你更想要找一本CSS参考书。网络上就有许多参考书，如O'Reilly的《CSS Pocket Reference》就是一本很好的手册。

问： **为什么要用另一种语言定义这些样式，而不用XHTML呢？既然我们用XHTML编写元素，那为什么不直接用XHTML写样式呢？**

答： 在后面几章，你将会发现使用CSS的一些明显的好处。在此先做个简要的回答：CSS比XHTML更适合指定样式的信息。使用一个小的CSS部件，就能对XHTML的设计产生明显的效果。你还会发现CSS更适合于处理多页面网页的样式。稍后你将会在本章看到它是如何工作的。

BRAIN POWER

假设在一个段落里有一个<em>元素。如果你改变段落的背景颜色，你是否认为也必须改变<em>元素的背景颜色使它与段落的背景颜色匹配？

在你的XHTML中引入CSS

好的，现在你已经知道一些CSS语法了。你知道如何选择一个元素然后写一个带属性和属性值的规则，但你还必须把CSS引入到一些XHTML中。首先，我们需要一些XHTML来引进它。在下面几章，我们将重新访问老朋友——Starbuzz和Tony，还有他的电动滑板车之旅——并使这些东西更美观。但是，你认为谁更迫切地想让他的站点第一个被修饰？当然是Head First 休闲室的伙计们。这是Head First 休闲室主页的XHTML。记得吧？上一章我们修改了一些东西使它升级为严格的XHTML（你是否或多或少记起一点？）。现在，我们将添加一些样式标记，这是将样式引入到你的网页中的最简单的方法。

但这未必是最好的方法。在本章后面我们将回顾这一方法并使用另一种方法。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en" >
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
    <title>Head First Lounge</title>

    <style type="text/css">


    </style>

  </head>
  <body>
    <h1>Welcome to the Head First Lounge</h1>
    <p>
       <img src="images/drinks.gif" alt="Drinks" />
    </p>
    <p>
       Join us any evening for refreshing
       <a href="beverages/elixir.html">elixirs</a>,
       conversation and maybe a game or two
       of <em>Dance Dance Revolution</em>.
       Wireless access is always provided;
       BYOWS (Bring your own web server).
    </p>
    <h2>Directions</h2>
    <p>
       You'll find us right in the center of downtown
       Webville. If you need help finding us, check out our
       <a href="about/directions.html">detailed directions</a>.
       Come join us!
    </p>
  </body>
</html>
```

这是我们感兴趣之处：<style>元素。

为了在你的XHTML中直接添加CSS样式，必须在<head>元素里添加样式开始和结束标记。

还有一个“text/css”样式类型。

而你的CSS规则将会处于这个位置。



在休闲室中添加样式

既然你已在你的XHTML中引入了**<style>**元素，你不妨在休闲室中添加一些样式以便熟悉如何书写CSS。这个设计可能无法令你获得“设计奖”，但你必须得有一个开始。

我们要做的第一件事是改变段落中文本的颜色（以此来匹配那些红色的休闲室沙发）。为此，我们将使用CSS **color**属性，如下：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">
  <head>
    <meta http-equiv="Content-Type"
          content="text/html; charset=ISO-8859-1" />
    <title>Head First Lounge</title>
    <style type="text/css">

      p {
          color: maroon;
      }

    </style>
  </head>
  <body>
    <h1>Welcome to the Head First Lounge</h1>
    <p>
      <img src="images/drinks.gif" alt="Drinks" />
    </p>
    <p>
      Join us any evening for refreshing
      <a href="beverages/elixir.html">elixirs</a>,
      conversation and maybe a game or two
      of <em>Dance Dance Revolution</em>.
      Wireless access is always provided;
      BYOWS (Bring your own web server).
    </p>
    <h2>Directions</h2>
    <p>
      You'll find us right in the center
      of downtown Webville. If you need
      help finding us, check out our
      <a href="about/directions.html">detailed directions</a>.
      Come join us!
    </p>
  </body>
</html>
```

这是用来指定段落字体颜色的规则。

我们选择<p>元素来进行设计。

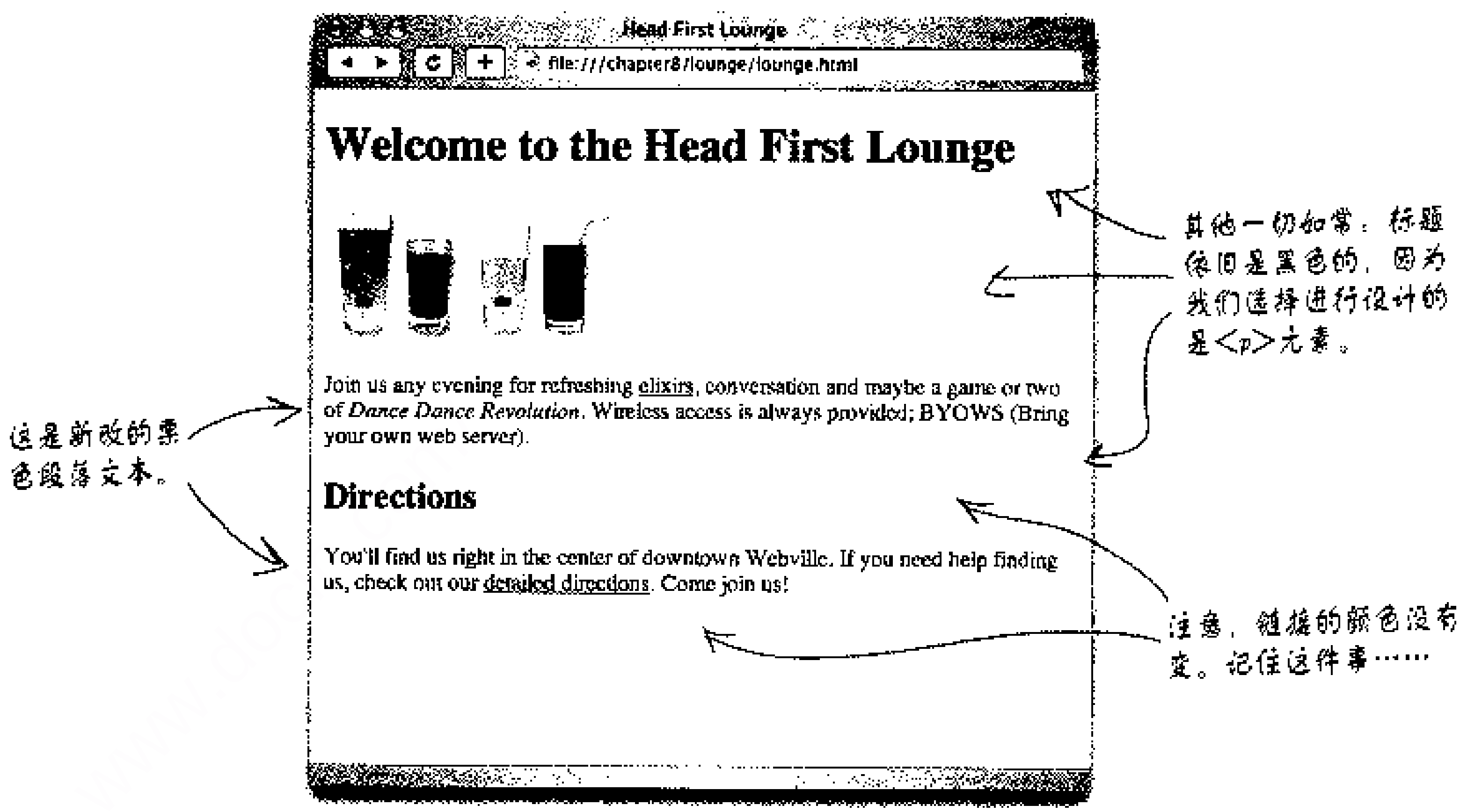
用来改变字体颜色的属性称为“color”（你也许会认为应该是“font-color”或“text-color”，但它不是）。

我们把文本设置为可爱的栗色，这正好与休闲室的沙发相配。

P选择符选择了XHTML中的所有段落。

查看样式：测试

继续学习并根据前面几页对“chapter8/lounge”文件夹里的“lounge.html”文件进行修改，保存，并在浏览器中重新加载你的网页。你将会看到那些段落文本的颜色已经变为栗色：

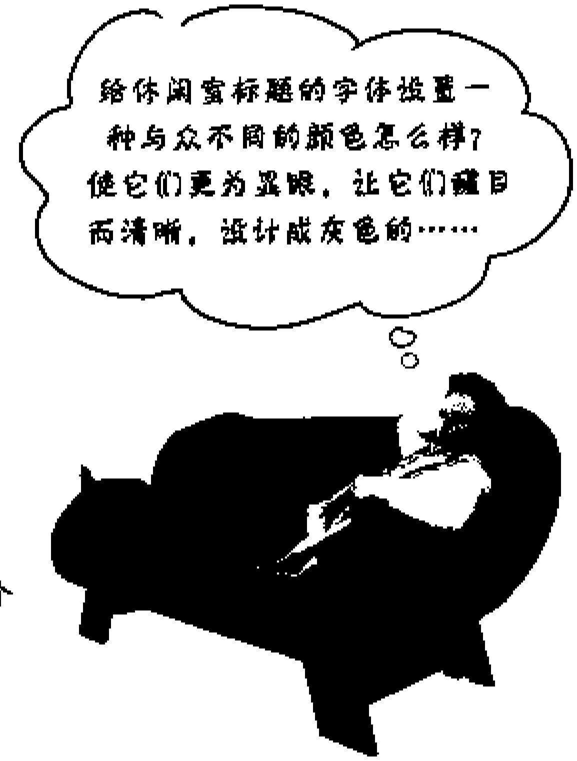


BRAIN POWER

如果你把**<p>**元素的**background-color**而不是**color**设置成栗色，那么会有什么结果呢？浏览器对网页的显示将如何变化？

为标题添加样式

现在，让我们对那些标题进行修饰。改变一下字体怎么样？让我们改变一下标题的字体和颜色：




```
h1 {
      font-family: sans-serif;
      color:       gray;
}

h2 {
      font-family: sans-serif;
      color:       gray;
}

p {
      color: maroon;
}
```

这是选择<h1>元素进行设计的规则并把字体设为sans-serif，把字体颜色设为灰色。稍后我们将就字体进行更深入的讨论。

这是对<h2>元素的另一个规则，功能与上面相同。

事实上，既然这些规则都完全一样，我们可以将它们整合在一块，如下：

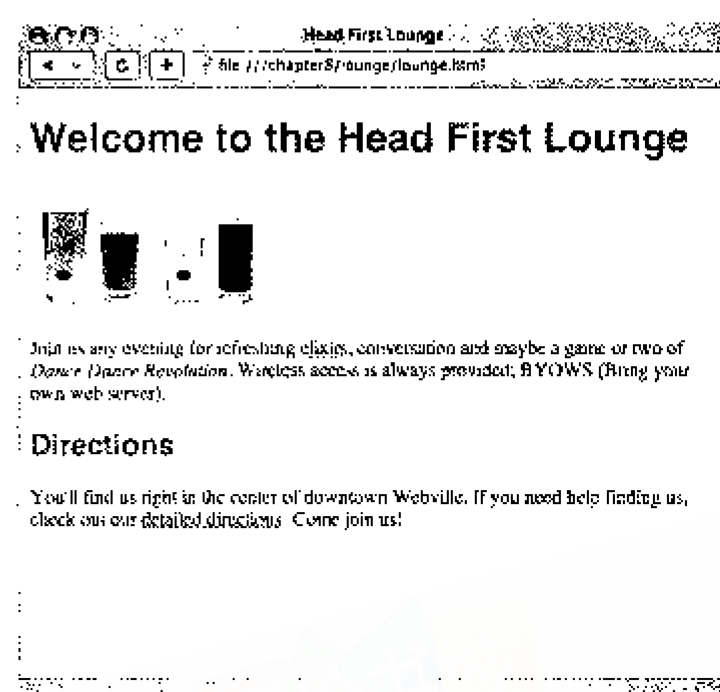
若想要为多个元素写同一个规则，只需在选择符之间加逗号，如"h1、h2"。

```
h1, h2 {
      font-family: sans-serif;
      color:       gray;
}

p {
      color: maroon;
}
```

测试……

添加这个新CSS到你的"lounge.html"文件并重新加载。你将会发现，你用同一个规则对**<h1>**标题和**<h2>**标题都进行了设置。



网页上的两个标题都被设置成了sans-serif字体和灰色。

给欢迎词加下划线

让我们进一步润色欢迎标题，给它加上下划线怎么样？那将使主标题更加显眼。下面是我们用来实现此功能的属性：

```
border-bottom: 1px solid black;
```

这个属性控制元素下划线的外观。

我们将把下划线设置成1像素宽的黑实线。

问题是，如果我们添加这个属性和属性值到“h1，h2”组合的规则里，我们得到的结果将是两个标题都有下划线：

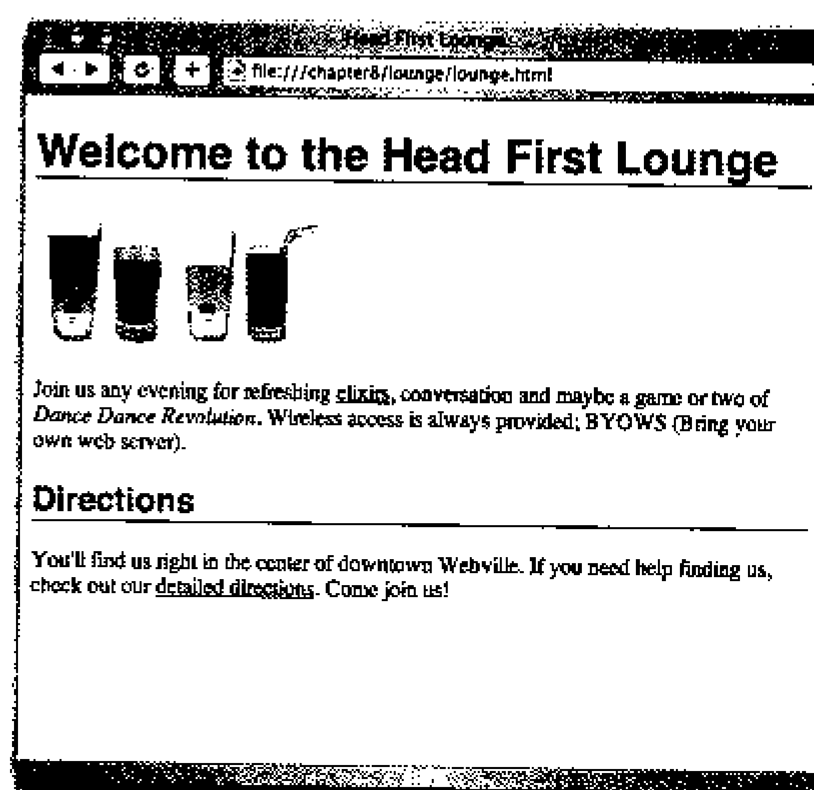
```
h1, h2 {
     font-family:   sans-serif;
     color:         gray;
     border-bottom: 1px solid black;
}

p {
     color: maroon;
}
```

这里我们添加一个属性来设置<h1>元素和<h2>元素的下划线。

如果我们这么做……

……将会使两个标题都有下划线。这不是我们的初衷。



那么，我们怎样才能保证下划线的效果只显示在**<h1>**元素上，而不对**<h2>**元素起作用呢？我们是否必须再分解规则呢？翻到下一页去看个究竟……

我们有这样的技术：指定只针对<h1>的规则

不必把“h1,h2”规则分开，只要再另外给“h1”加一条规则，在里面样式化边框就行了。

```
h1, h2 {
    font-family:    sans-serif;
    color:          gray;
}

h1 {
    border-bottom: 1px solid black;
}

p {
    color: maroon;
}
```

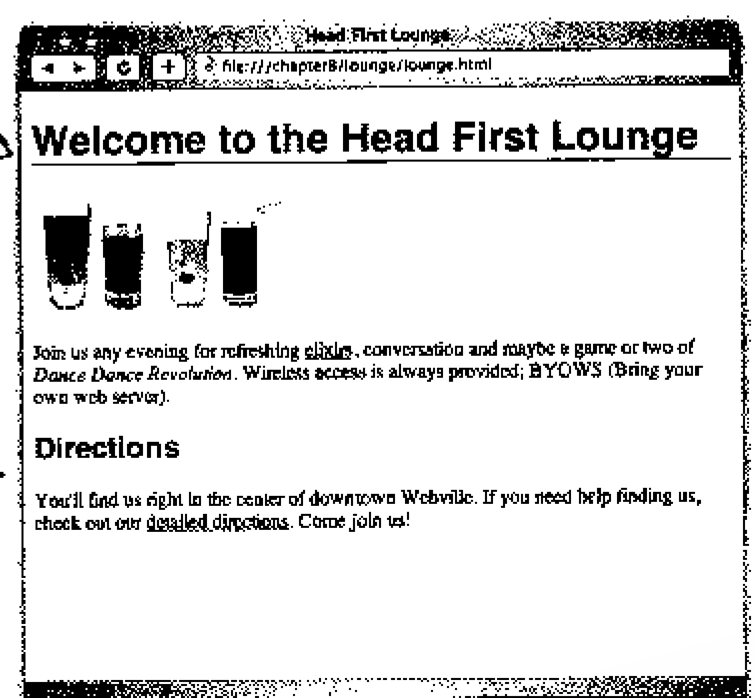
第一个规则保留原样。我们依然要使用组合的规则来设置<h1>和<h2>的字体类型和颜色。

但现在我们的第二个规则只给<h1>添加另一个属性：下划线属性。

另一个测试……

更改你的CSS并重新加载网页。你将会看到新规则在欢迎标题上添加了下划线，而加了下划线的标题比之前更显眼。

这是黑色下划线。



正如我们所希望的，这里没有下划线。

there are no Dumb Questions

问： 如果你给一个元素写多个规则，那结果会如何呢？

答： 你可以随心所欲地给一个元素写若干规则。每个规则都会在前面添加样式信息。

大体上，你试着把元素之间共有的样式组合在一起，就如我们对<h1>和<h2>所做的那样，而把特定于一个元互的样式写在另一个规则里面，如我们给主标题加入下划线样式。

问： 这种方法的优点是什么呢？对每个元素分开设计，不是更有利于我们区分各个元素的样式吗？

答： 不见得。如果你把共有的样式结合在一起，如果要改动，你只需修改一个规则。如果你把它们拆开，那你就不得不修改每一个规则，这样反而容易出错。

问： 为什么我们要用底部边框加下划线？不是已经有一个下划线样式可以对文本进行修饰吗？

答： 问得好。有一个下划线样式可以对文本进行修饰，我们可以用它而不用底部边框。然而，这两个样式在网页上有明显不同的效果：如果是底部边框，那么标题下的线将延伸到网页边沿，而下划线只在文本下方显示。设置文本下划线的属性叫做text-decoration并且它的值为“underline”。动手试试并检验一下两者的区别。

那么，选择符如何工作？

你已经知道如何选择一个元素来为它添加样式了，如下：

我们称之为选择符。

```
h1 {
      color: gray;
}
```

这个样式适用于选择符（此时，为<h1>元素）选择的元素。

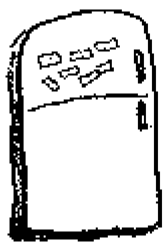
或者，如何选择多个元素，像这样：

另一个选择符，这个样式适用于<h1>元素和<h2>元素。

```
h1, h2 {
      color: gray;
}
```

CSS允许你定义各种选择符来决定你的样式适用于哪些元素。知道如何使用选择符是掌握CSS的第一步，而要做到这点，你必须了解你正在设计的XHTML的组成元素。如果对于XHTML由哪些元素组成都没有弄清楚，那你如何选择元素进行设计呢？如何把它们联系起来呢？

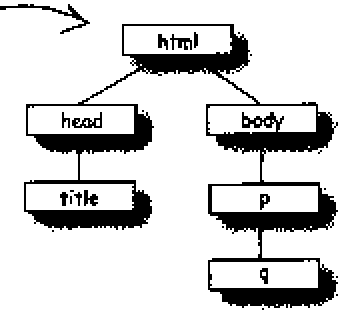
记住休闲室XHTML的结构图吧！接着我们将继续深入了解选择符。

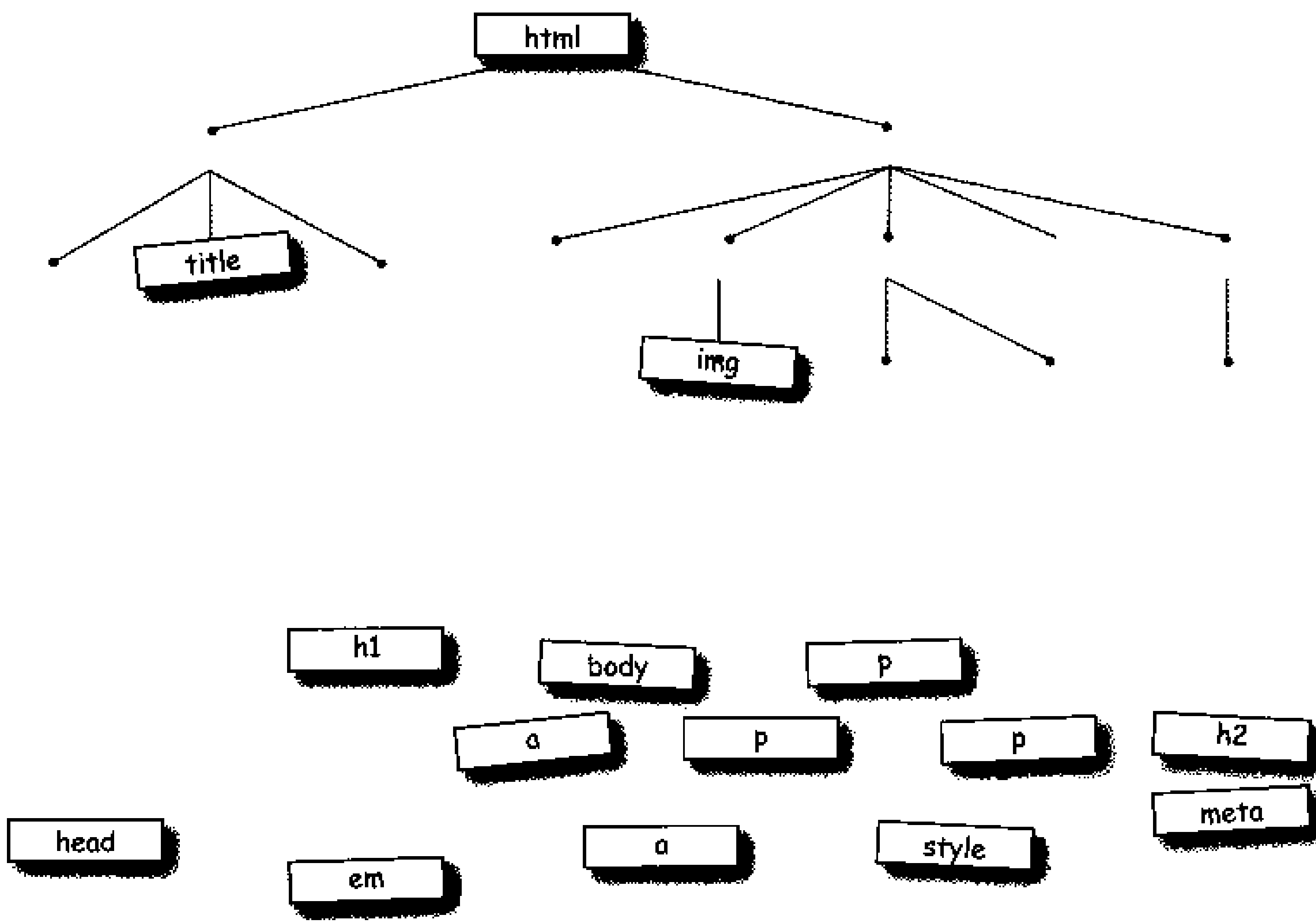


标记帖

记得我们在第3章画的HTML元素图吗？我们将给休闲室的主页画元素图。下面是你完成该图表所必需的元素帖。依据休闲室的XHTML（如右图），完成下面的树图，我们已经加上了几个标记帖。本章后面有答案。

像这样。





Head First休闲室的XHTML。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
    <title>Head First Lounge</title>

    <style type="text/css">
      h1, h2 {
               font-family:   sans-serif;
               color:         gray;
      }

      h1 {
               border-bottom: 1px solid black;
      }

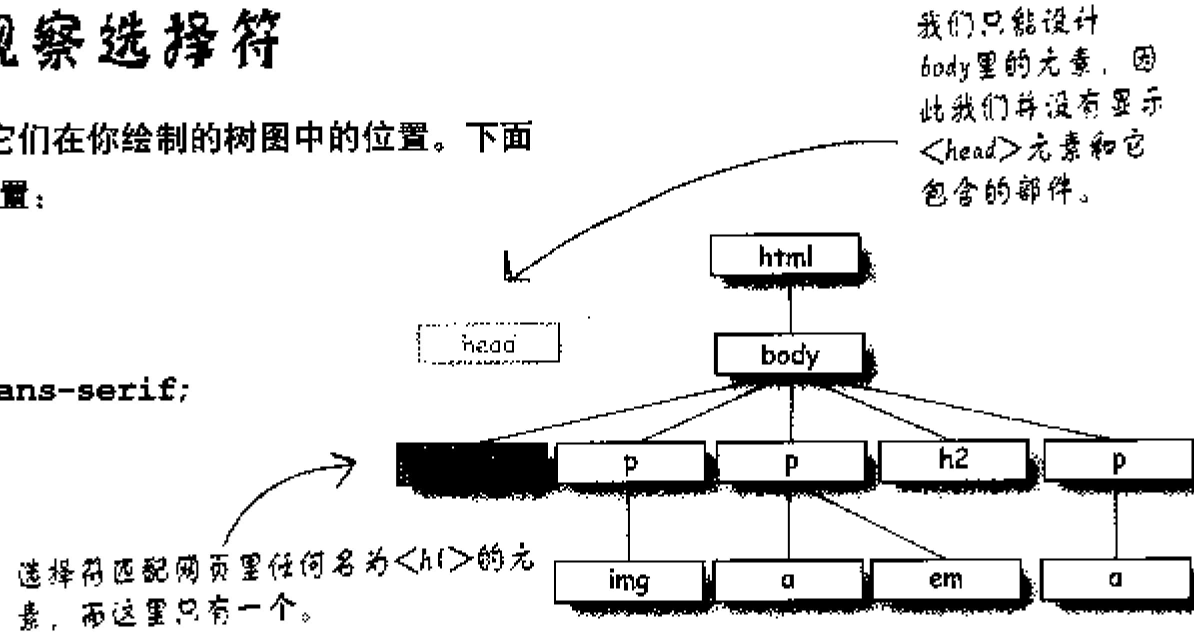
      p {
               color: maroon;
      }
    </style>

  </head>
  <body>
    <h1>Welcome to the Head First Lounge</h1>
    <p>
      <img src="images/drinks.gif" alt="Drinks" />
    </p>
    <p>
      Join us any evening for refreshing
      <a href="beverages/elixir.html">elixirs</a>,
      conversation and maybe a game or two
      of <em>Dance Dance Revolution</em>.
      Wireless access is always provided;
      BYOWS (Bring your own web server).
    </p>
    <h2>Directions</h2>
    <p>
      You'll find us right in the center of downtown
      Webville. If you need help finding us, check out our
      <a href="about/directions.html">detailed directions</a>.
      Come join us!
    </p>
  </body>
</html>
```

通过图解仔细观察选择符

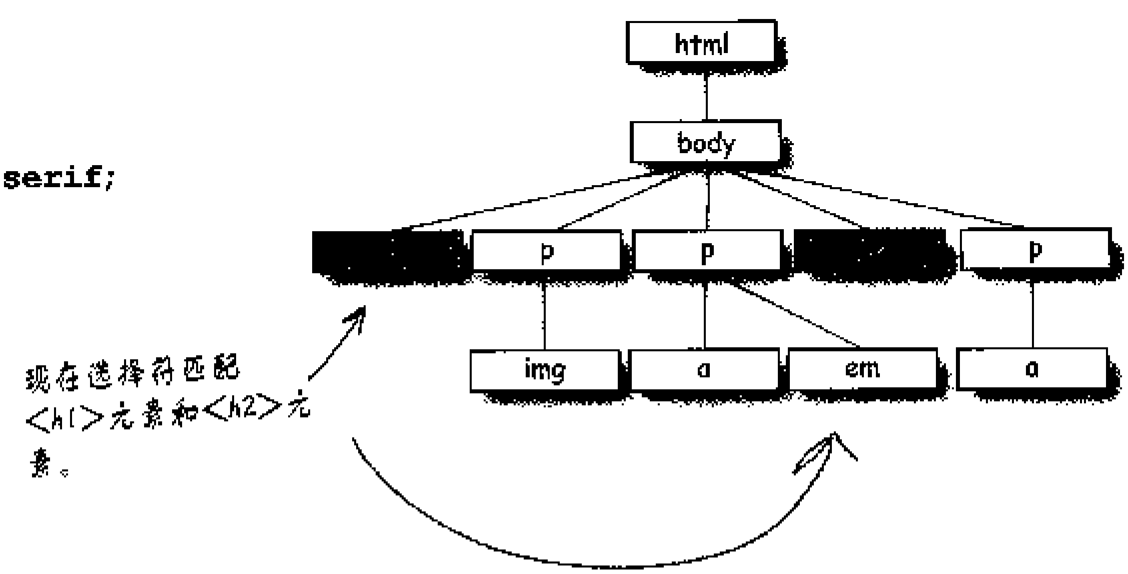
让我们举出一些选择符看看它们在你绘制的树图中的位置。下面是“h1”选择符在图中的位置：

```
h1 {
    font-family: sans-serif;
}
```



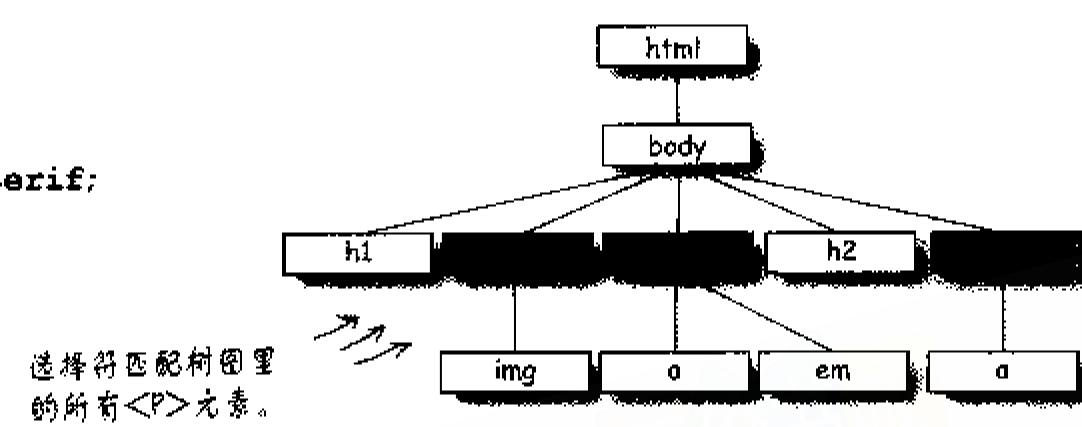
“h1, h2” 选择符是以下这样的：

```
h1, h2 {
    font-family: sans-serif;
}
```



如果用“p” 选择符，那么就是以下这样的：

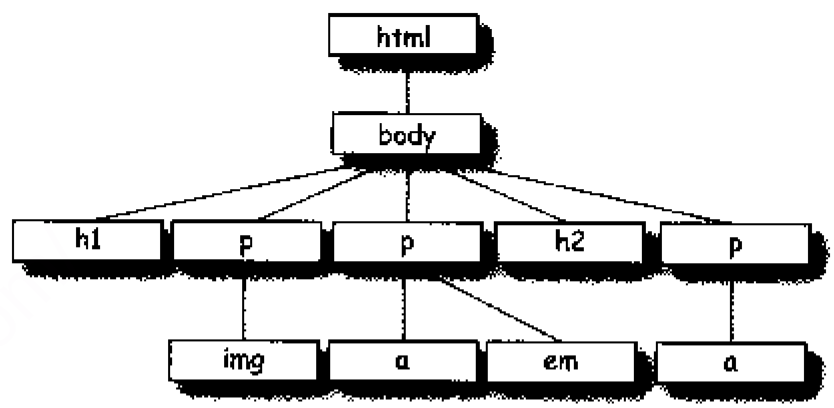
```
p {
    font-family: sans-serif;
}
```



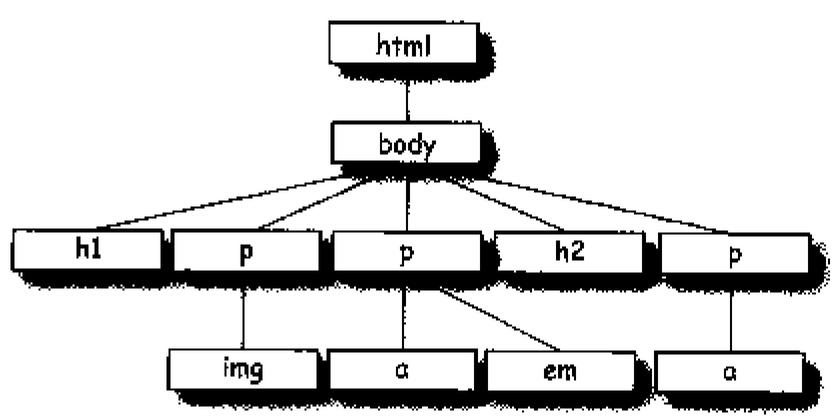
Sharpen your pencil

给选择符选择的元素着色：

```
p, h2 {
     font-family: sans-serif;
}
```




```
p, em {
     font-family: sans-serif;
}
```



五分钟之谜



蛮力对样式的案件

上次我们在第4章离开RadWebDesign时，他们已经取消了所有的演示，并放弃了RobotsRUs的业务。CorrectWebDesign全面接管了RobotsRUs站点并将在本月该站点启用前敲定所有的事务。但是，你也记得RadWebDesign决定致力于他们的XHTML和CSS吧。他们决定使用严格的XHTML和样式表重做RobotsRUs站点，这可让他们在接到其他咨询工作之前获得一些经验。

就好像是命中注定的一样，RobotsRUs的站点在启动之前，情况再次发生：因为一个紧急信息，RobotsRUs找到了CorrectWebDesign，“我们正在改变网页的整体外观,如网页的文本颜色、背景颜色、字体等。”关键在于，这个站点几乎包含一百个页面，因此CorrectWebDesign回应说他们得花几天时间来改写这个网站。CEO说：“我们没有那么多时间。”不得已，CEO只好求助于RadWebDesign。“虽然上个月你们把工作搞得一团糟，但我们现在确实需要你们的帮助。你们能帮助CorrectWebDesign修改网站的样式吗？”RadWebDesign说他们能做得比这更好；事实上，他们在一个小时内就可以完成这工作。

RadWebDesign如何从菜鸟摇身一变成为网页高手呢？他们如何能够以不可思议的速度改变一百个网页的样式呢？



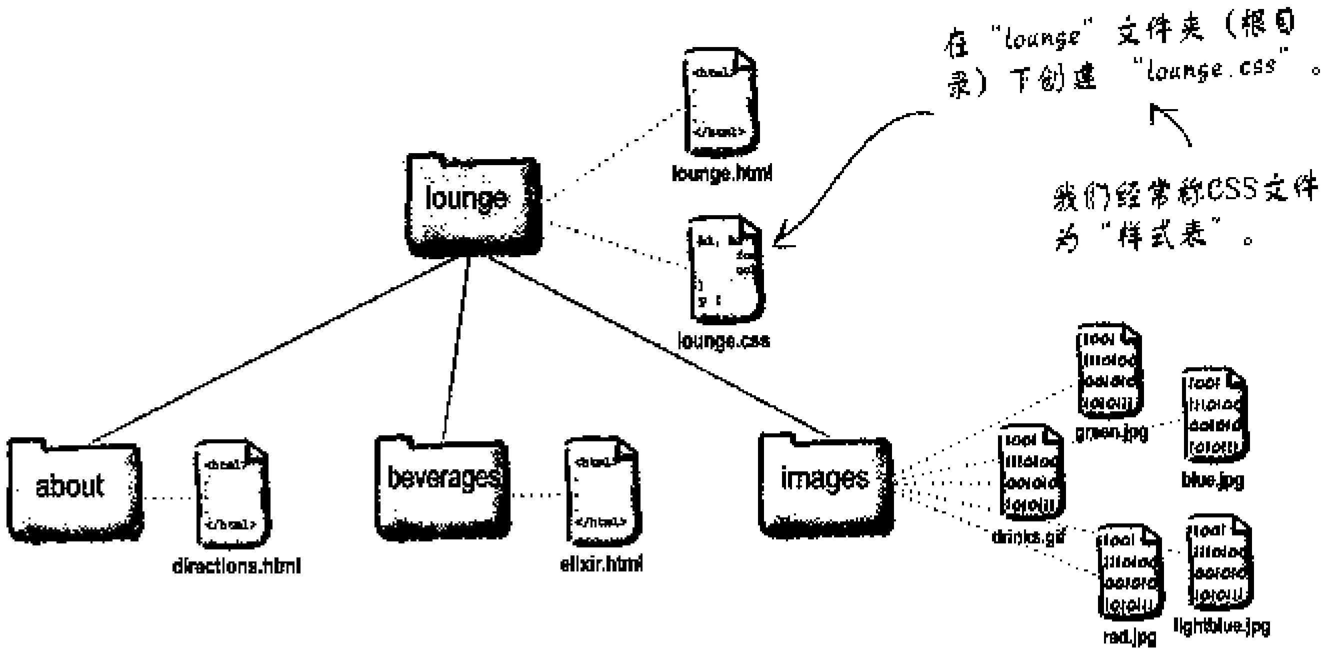
对elixir和directions页面进行同休闲室一样的设计

我们已经对“lounge.html”进行了设计，但elixir和directions页面呢？它们必须与主页有一个协调的外观。很容易……只需复制样式元素和规则到这两个文件，是吗？没那么简单。如果这么做，那么当你要改变一个网站的样式时，你就不得不修改每一个文件——而这不是你所希望的。很幸运，有一个更好的方法。步骤如下：

1. 取出“lounge.html”里的规则并以一个名为“lounge.css”的文件保存它们。
2. 创建一个从“lounge.html”到这个文件的外部链接。
3. 给“elixir.html”和“directions.html”也创建同样的链接。
4. 好好测试这三个文件。

创建“lounge.css”文件

创建一个名为“lounge.css”的文件来包含对Head First休闲室网页进行修饰的样式规则。为此，在你的文本编辑器里创建一个新文本文件并命名为“ lounge.css”。



现在，用键盘输入，或者从你的“lounge.html”文档中复制CSS规则粘贴到“lounge.css”中。删除“lounge.html”里的规则。

注意，不要连同**<style>**和**</style>**标记也复制了，因为“lounge.css”只包含CSS，而不包括XHTML。

```
h1, h2 {
   font-family: sans-serif;
   color: gray;
}

h1 {
   border-bottom: 1px solid black;
}

p {
   color: maroon;
}
```

你的“lounge.css”文件应该像这样，记住，没有<style>标记！

把“lounge.html”链接到外部样式表

现在，我们必须告诉浏览器，让它用外部样式表里的样式修饰网页。我们可以用一个叫做**<link>**的XHTML元素实现这个功能。下面是在你的XHTML里使用**<link>**元素的例子：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">
  <head>
    <meta http-equiv="Content-Type"
          content="text/html; charset=ISO-8859-1" />
    <title>Head First Lounge</title>
    <link type= "text/css" rel="stylesheet" href= "lounge.css" />
    <style type="text/css">
    </style>

  </head>
  <body>
    <h1>Welcome to the Head First Lounge</h1>
    <p>
      <img src="drinks.gif" alt="Drinks" />
    </p>
    .
    .
    .
    </p>
  </body>
</html>
```

这是链接到外部样式表的XHTML。

你不再需要<style>元素了——删掉它。

其他的XHTML不变。

靠近XHTML一点

既然以前我们没有接触过**<link>**元素，那就让我们详细地了解一下它吧：

```
<link type="text/css" rel="stylesheet" href="lounge.css" />
```

使用link元素来链接外部信息。

信息的类型是“text/css”，也就是一个CSS样式表。

而样式表定位于这个href（现在我们使用的是相对链接，但它可以是一个完整的URL）。

rel属性指明XHTML文件和你要链接的东西之间的关系。我们要链接到一个样式表，因此我们将属性值设为“stylesheet”。

<link>是一个空元素。

把“elixir.html”和“directions.html”链接到外部样式表

就如你对“lounge.html”做的那样，现在你将要链接“elixir.html”和“directions.html”文件。唯一要记住的是：“elixir.html”文件在“beverages”文件夹里，而“directions.html”在“about”文件夹里，因此它们都必须使用相对路径“../lounge.css”。

因此，你要做的就是把下面的`<link>`元素添加到这两个文件中：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
                      "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
    <title>Head First Lounge Elixirs</title>
    <link type="text/css" rel= "stylesheet" href="../lounge.css" />
  </head>
  <body>
  .
  .
  .
  </body>
</html>
```

这是“elixir.html”，只需添加<link>行。

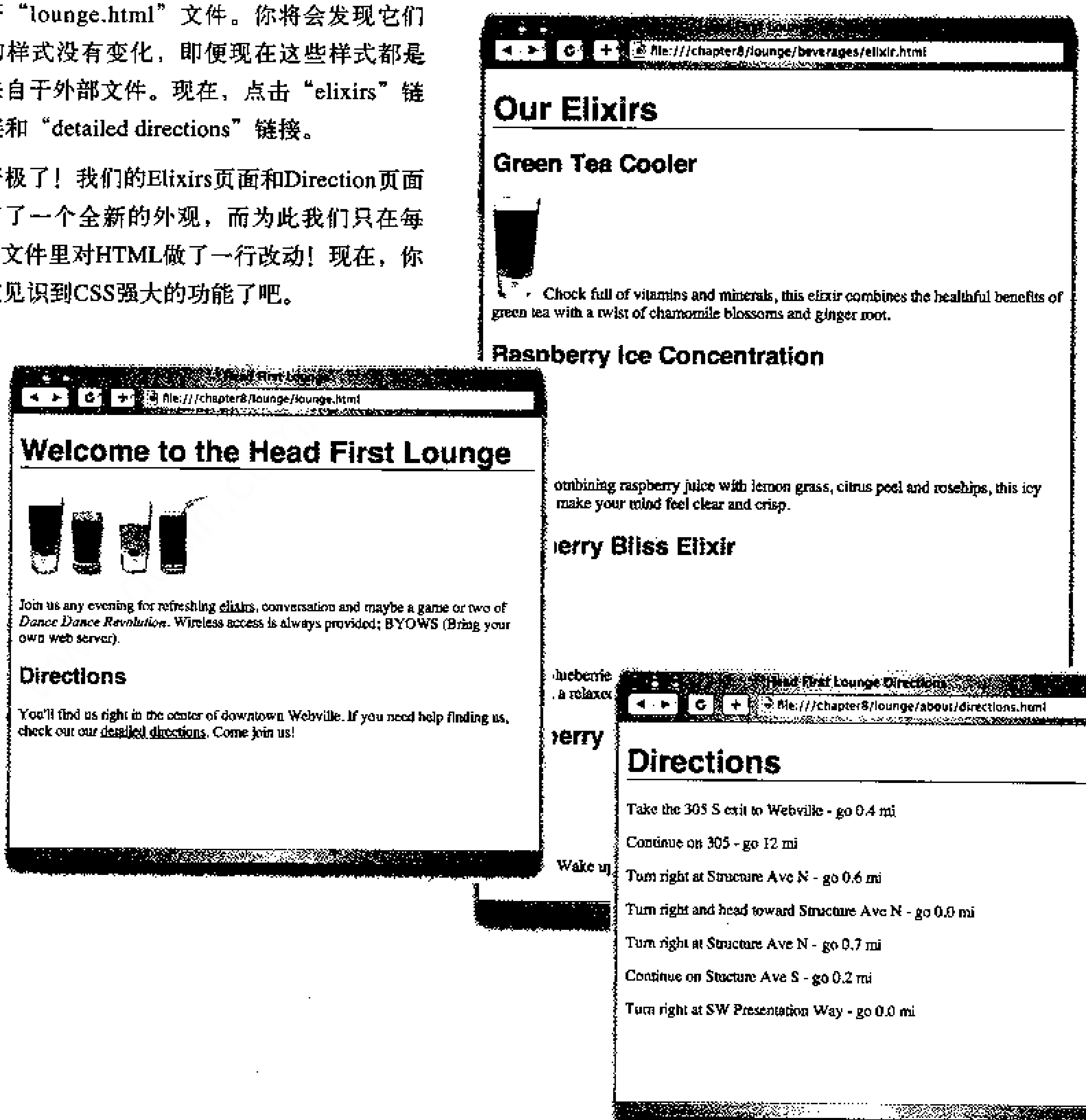
```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
                      "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
    <title>Head First Lounge Directions</title>
    <link type= "text/css" rel="stylesheet" href="../lounge.css" />
  </head>
  <body>
  .
  .
  .
  </body>
</html>
```

对“directions.html”做相同的操作，在这儿添加<link>行。

调试整个休闲室……

保存这些文件，然后用浏览器打开“lounge.html”文件。你将会发现它们的样式没有变化，即便现在这些样式都是来自于外部文件。现在，点击“elixirs”链接和“detailed directions”链接。

好极了！我们的Elixirs页面和Direction页面有了一个全新的外观，而为此我们只在每个文件里对HTML做了一行改动！现在，你该见识到CSS强大的功能了吧。



解开五分钟之谜



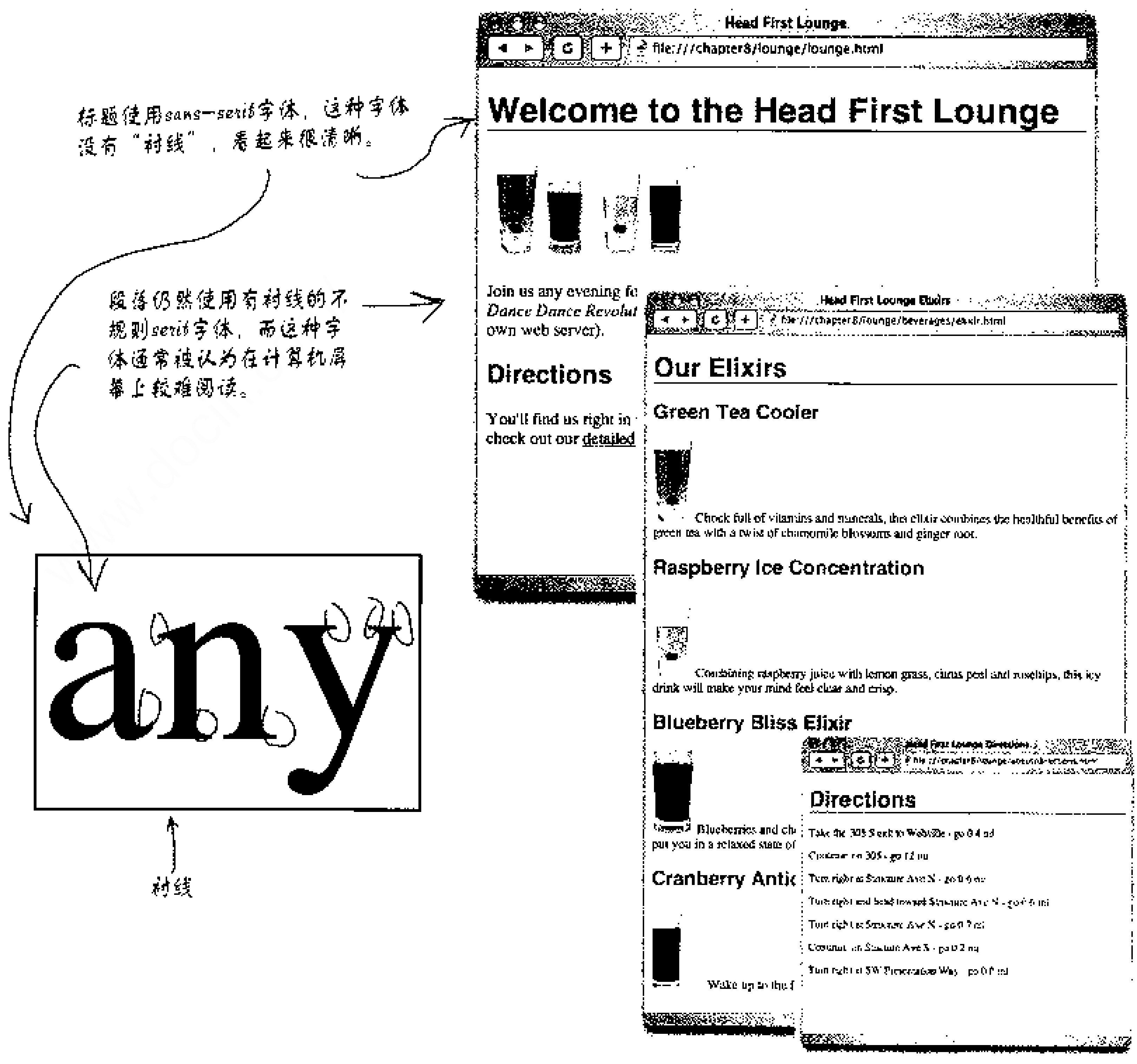
蛮力对样式的案件

RadWebDesign是怎样变为网页高手的？或者，我们应该先问，为什么一向“不犯错”的CorrectWebDesign公司这次会把事情搞得一团糟呢？问题的根源是CorrectWebDesign使用了1998年的技术创建RobotsRUs网页。他们在HTML里正确地添加了他们的style规则（每次都是复制并粘贴它们）。但糟糕的是他们使用了许多旧的元素，如**<font>**和**<center>**，而现在这些旧元素已经不被接受了。因此，当要求更改外观的时候，就意味着要着手每个网页并对CSS进行修改。更糟的是，这也意味着必须遍历HTML来修改元素。

相比之下，RadWebDesign使用的是严格的XHTML 1.0，因此他们的网页里没有旧的HTML表达式，并且他们使用一个外部样式表。结果呢？为了修改整个站点的外观，他们只需访问外部样式表并对CSS做一些修改，这在几分钟内就可以轻易完成了，而不需花费几天时间。他们甚至有时间去测试多种设计方案，并在站点启动前用三种CSS版本来检查它。令人吃惊的是，RobotsRUs的CEO不仅许诺给他们更多的业务，还许诺给他们一套最新的自动设备。

Sharpen your pencil

既然你已经有了一个外部样式文件（或者“样式表”），就用它将所有段落的字体改为“sans-serif”以使之与标题匹配。记住，更改字体样式的属性是“font-family”，而设置sans-serif字体的属性值是“sans-serif”。答案在下一页。



Sharpen your pencil

答案

```
h1, h2 {
        font-family:  sans-serif;
        color:        gray;
}

h1 {
        border-bottom: 1px solid black;
}

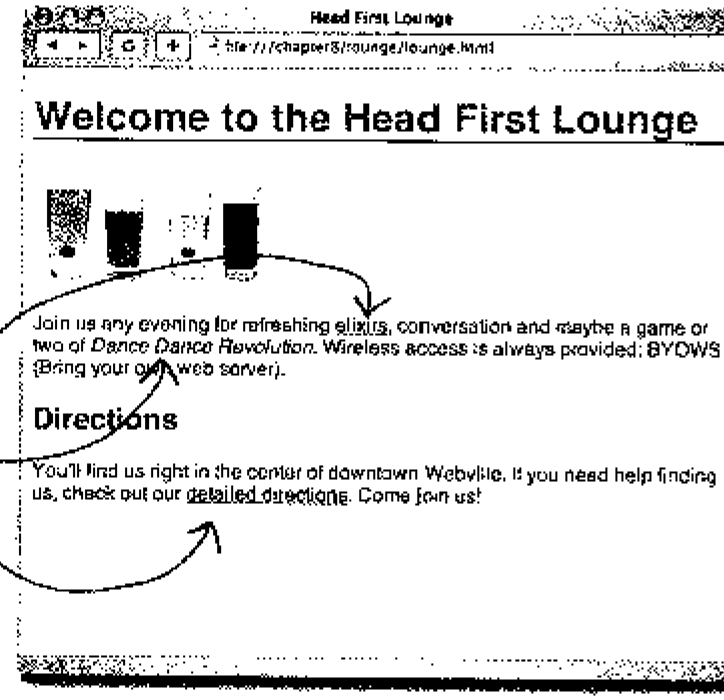
p {
        font-family:  sans-serif;
        color:        maroon;
}
```

只需在"lounge.css"文件的段落规则中添加一个font-family属性。



了解继承……

你是否注意到，当你向“p”选择符添加**font-family**属性时，存在于**<p>**元素中的元素的字体也会受到影响。让我们详细看一下：



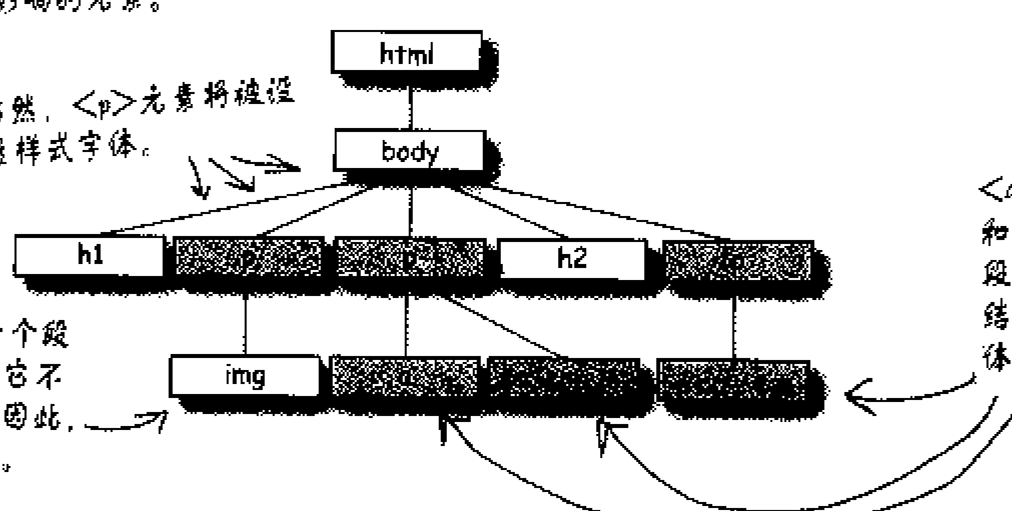
<p>中的元素从<p>元素继承了font-family样式

就如你能因父母的遗传拥有蓝眼睛和棕色的头发一样，元素也可以从它的父母那儿继承样式。如下面的情形，**<a>**元素和**<em>**元素从**<p>**元素继承了**font-family**样式，而**<p>**元素是它们的父元素。这意味着改变段落的样式将导致段落里元素的样式改变，是吗？是的，不然的话，你将不得不为整个站点里所有段落的每个行元素添加CSS规则……而这样的确很不方便。

不是所有的样式都能被继承的。只是一部分，如字体。

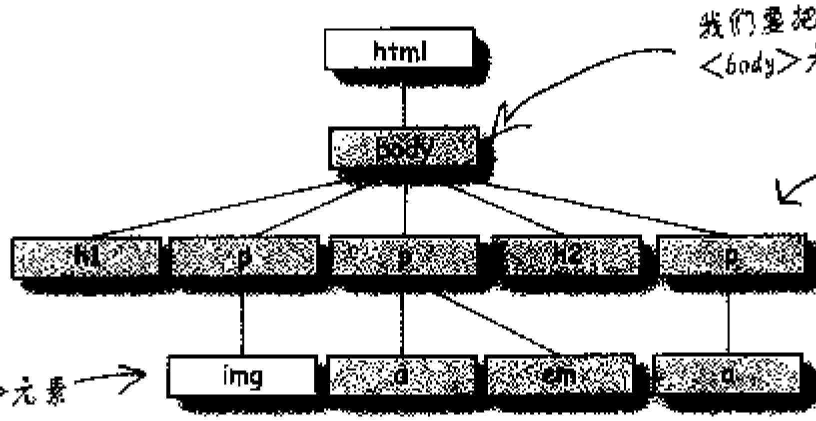
更不要说易出错、乏味和费时了。

让我们看一下XHTML树以便了解继承是如何起作用的：



如果我们把字体属性上移会有什么结果？

如果大部分的元素都继承**font-family**属性，那我们把字体属性上移到**<body>**元素，结果会怎样？属性将会对**<body>**元素的所有子孙结点生效，改变它们的字体。



还等什么呢……行动吧

打开你的“lounge.css”文件并添加选择**<body>**元素的新规则。然后从标题规则和段落规则中删除**front-family**属性，因为你不需要它们了。

下面是你要做的事：

```
body {
        font-family:  sans-serif;
}

h1, h2 {
        font-family:  sans-serif;
        color:        gray;
}

h1 {
        border-bottom: 1px solid black;
}

p {
        font-family:  sans-serif;
        color:        maroon;
}
```

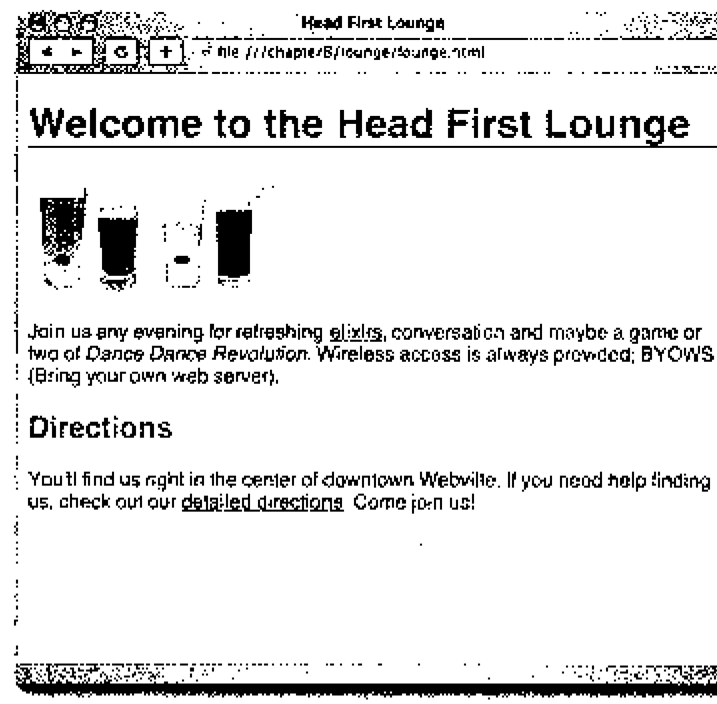
首先，添加一个选择<body>元素的新规则。然后给它添加字体属性并将值设为sans-serif。

然后，从h1、h2规则和p规则中把字体属性删除。（h1, h2 和 p 规则中的 font-family 行已被划掉。）

测试你的新CSS

像平常一样，继续深入并在“lounge.css”样式表里做些修改，保存，重新加载“lounge.html”页面。你不可能期待什么变化，因为样式是一样的。只有规则不同而已。但你会觉得你的CSS比以前好，因为现在你可以添加新元素到你的网页，而它们会自动继承sans-serif字体。

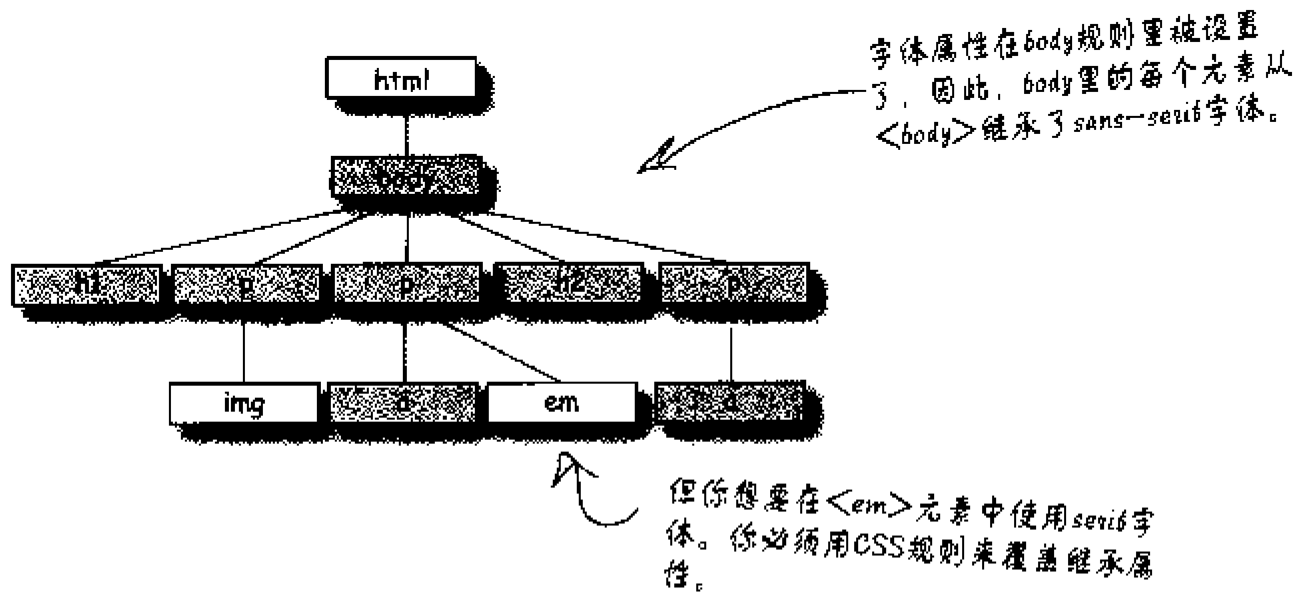
奇怪，真奇怪。看起来没有什么不同，这确实是我们希望的结果，不是吗？你所做的就是将字体属性上移到body规则并让所有的元素继承这个属性。





继承的覆盖

你已经通过把`font-family`属性上移到body从而设置了整个网页的字体。但如果你不想在所有的元素中都使用sans-serif字体呢？比如，你想在**<em>**元素中使用serif字体。



你可以为**<em>**元素提供一个具体的规则来覆盖它继承的属性。下面是如何给**<em>**元素添加一个规则来覆盖在body里定义的字体属性：

```
body {
        font-family:  sans-serif;
}

h1, h2 {
        color:        gray;
}

h1 {
        border-bottom: 1px solid black;
}

p {
        color:        maroon;
}

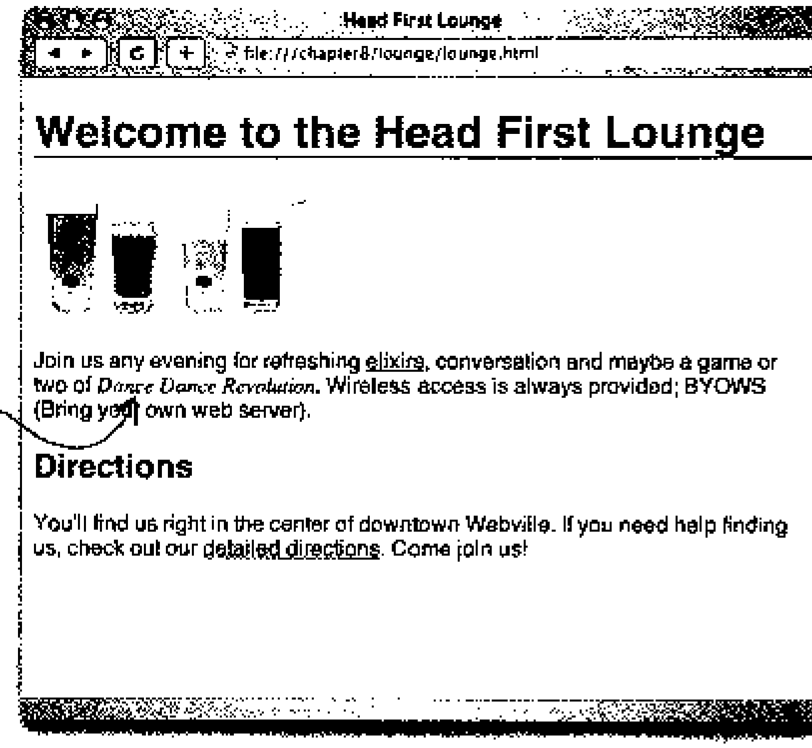
em {
        font-family: serif;
}
```

添加一个选择了em并设字体属性值为serif的新规则，以此来覆盖从body继承的字体属性。

测试

在CSS中给<em>添加一条规则，并将**font-family**属性值设置为**serif**，然后重新加载你的"lounge.html"页面：

注意"Dance Dance Revolution"文本，这是<em>元素里的文本，现在它用的是serif字体。



一般来说，像这样在一个段落的中间改变字体并非好主意，因此，如果你已经检验完毕，把你的CSS改回原样吧（没有<em>这个规则）。

there are no Dumb Questions

问：当我覆盖继承值时，浏览器如何知道应该对<em>元素执行哪一条规则？

答：在CSS里，被应用的规则总是最具体的。因此，如果<body>元素有一个规则，<em>元素也有一个更具体的规则，那浏览器一般会执行更具体的规则。稍后我们将更详细讨论你如何知道哪一个规则是更具体的规则。

问：我如何知道哪些CSS属性可被继承，哪些不能被继承呢？

答：这时一本好的参考书就派上用场了，如O'Reilly的《CSS Pocket Reference》。大体上，影响你文本外观的样式，如字体颜色（颜色属性）、字体（如你刚看到的），还有其他和字体有关的属性，如字体大小、字体宽度（如黑体）和字体样式如（斜体），都可以被继承。其他的属性（比如边框）不被继承，这很容易理解吧？你要给你的<body>元素加边框并不意味着你想要给所有的元素加边框。很多情况下你可以根据自己的理解（或者验证一下），然后，随着你对各种属性和它们的作用越来越熟悉，你就能掌握这方面的知识了。

问：如果我不再使用一个继承属性，我能将它覆盖吗？

答：是的。你总能用一个更具体的选择符覆盖一个来自于父结点的属性。

问：这程序很复杂。我可以添加注释来提醒自己所写规则的作用吗？

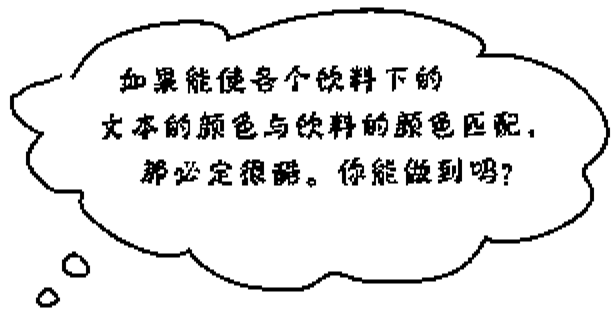
答：可以。只需在首尾分别添加/*和*/，就可以在你的CSS里添加注释。例如：

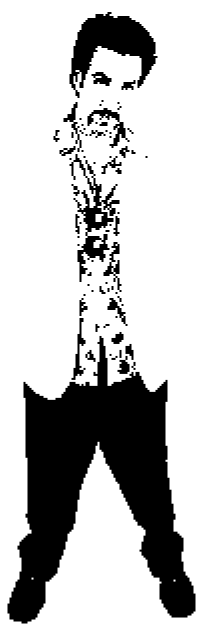
```
/* this rule selects all
paragraphs and colors them
blue */
```

注意，一个注释可以跨越多行。你可以在CSS周围添加注释，而浏览器将会忽略它，如：

```
/* this rule will have no
effect because it's in a
comment

p { color: blue;} */
```

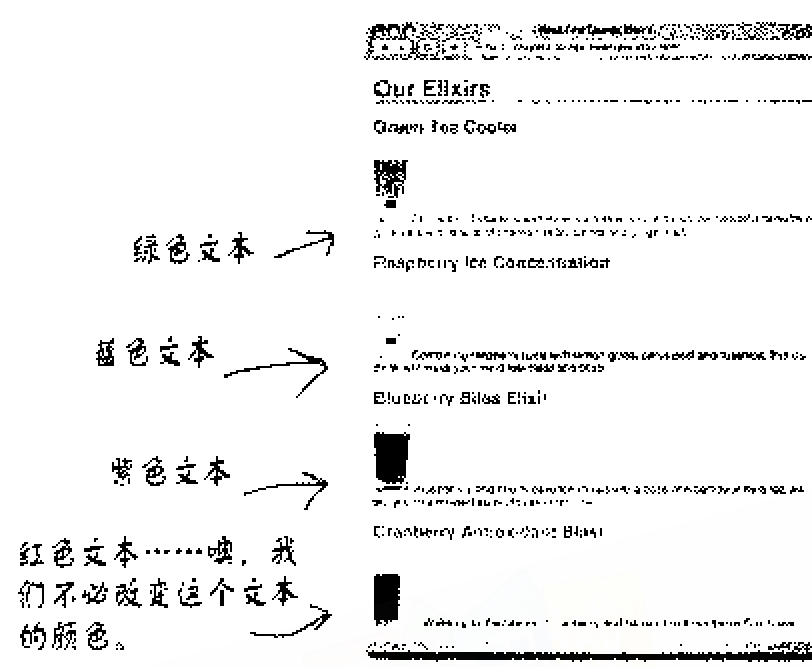




我们不赞同你的审美观，但你是我们的客户，客户至上。

你可以分别对那些段落进行修饰，使文本的颜色与饮料的颜色匹配吗？问题的关键是给所有的<p>元素各提供一条带有“p”选择符的规则，那么，你如何单独选择这些段落呢？

这就是引入类的原因。使用XHTML和CSS，你可以定义一个元素类，然后提供样式给属于那个类的任何元素。那么，类究竟是什么呢？把它比喻成俱乐部——某人开办了一个“绿茶”俱乐部，而且如果你加入了，就可以获得相应的权利和责任，比如遵循他们的样式标准。让我们创建类并看看它是如何工作的。



在"elixir.html"中添加类

打开"elixir.html"文件并找到"酷爽绿茶"段落。这是我们要改成绿色的文本。我们要做的就是添加这个<p>元素到一个名为**greentea**（绿茶）的类中。如下所示：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
    <title>Head First Lounge Elixirs</title>
    <link type="text/css" rel="stylesheet" href="../lounge.css" />
  </head>
  <body>
    <h1>Our Elixirs</h1>
    <h2>Green Tea Cooler</h2>
    <p class="greentea">
          <img src="../images/green.jpg" />
          Chock full of vitamins and minerals, this elixir
          combines the healthful benefits of green tea with
          a twist of chamomile blossoms and ginger root.
    </p>
    <h2>Raspberry Ice Concentration</h2>
    <p>
          <img src="../images/lightblue.jpg" />
          Combining raspberry juice with lemon grass,
          citrus peel and rosehips, this icy drink
          will make your mind feel clear and crisp.
    </p>
    <h2>Blueberry Bliss Elixir</h2>
    <p>
          <img src="../images/blue.jpg" />
          Blueberries and cherry essence mixed into a base
          of elderflower herb tea will put you in a relaxed
          state of bliss in no time.
    </p>
    <h2>Cranberry Antioxidant Blast</h2>
    <p>
          <img src="../images/red.jpg" />
          Wake up to the flavors of cranberry and hibiscus
          in this vitamin C rich elixir.
    </p>
  </body>
</html>
```

要把元素添加到一个类中，只需添加一个带有类名（如"greentea"）的"class"属性。

既然描述绿茶的段落属于**greentea**类，那么你只需提供一些规则来设置那个元素类就行了。

创建类的选择符

要选择一个类，你可以如下编写选择符：

```
p.greentea {
    color: green;
}
```

首先是p选择符。

然后用一个“.”来指定一个类。

最后是类的名称。

这是规则……将绿茶类里所有段落的文本设置为绿色。

这个选择符选择绿茶类里的所有段落。

现在你已经掌握选择属于某个类的<p>元素了。你必须做的是添加**class**属性到你想要设置为绿色的<p>元素中，并且应用这条规则。试一下：打开你的“lounge.css”文件并添加**p.greentea**类选择符。

```
body {
      font-family: sans-serif;
}

h1, h2 {
      color: gray;
}

h1 {
      border-bottom: 1px solid black;
}

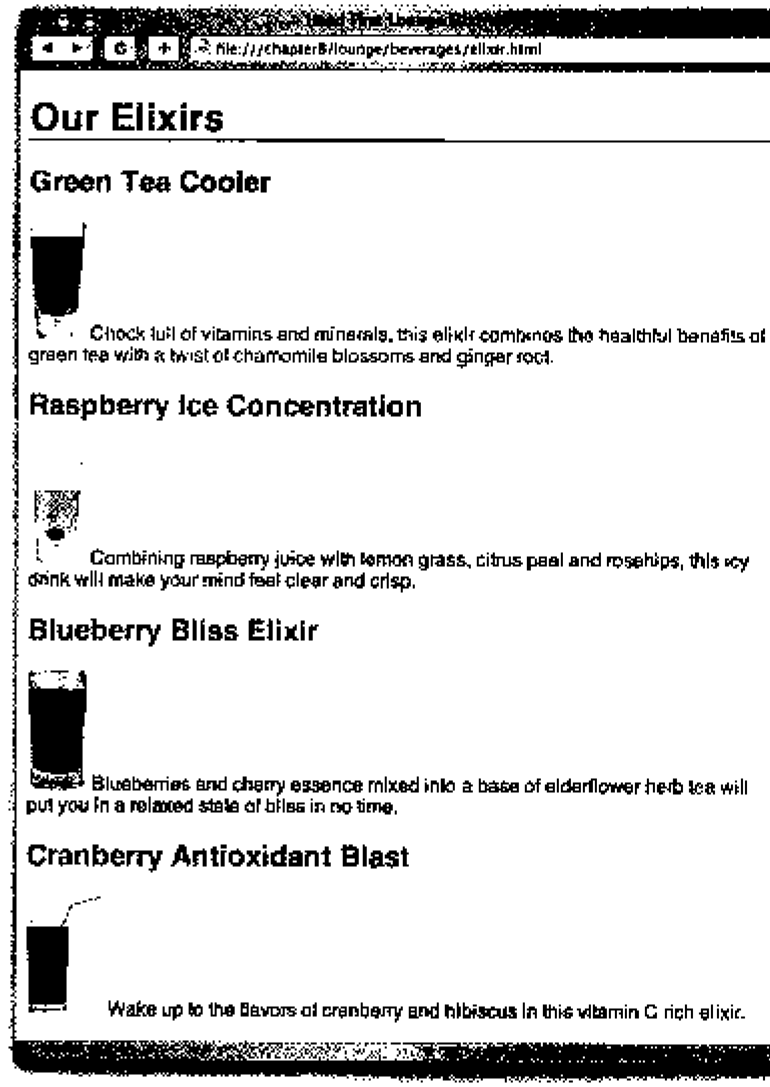
p {
      color: maroon;
}

p.greentea {
      color: green;
}
```

测试绿茶类

保存，并重新加载，测试一下你的新类。

这是应用于段落的绿茶类。现在字体是绿色的并且与酷爽绿茶的颜色匹配。或许这个样式还不错。



Sharpen your pencil

轮到你了：添加两个类——“raspberry”和“blueberry”到“elixir.html”里的相应段落中，然后编写样式分别将文本设置为蓝色和紫色。raspberry属性的值设为“blue”，而blueberry的属性值设为“purple”。将这些规则置于你的CSS文件的底部，放在绿茶规则下面：先是raspberry，跟着是blueberry。

你可能会想：raspberry怎么会是蓝色的呢？如果raspberry Kool-aid是蓝色的，那对我们来说当然很好。严格地说，当你混合一箱蓝莓时，它们确实更接近紫色，而非蓝色。按我们的要求去做吧。

进一步了解类

你已经编写了一条规则，而这条规则使用**greentea**类来将类里的段落改为“绿色”：

```
p.greentea {
        color: green;
}
```

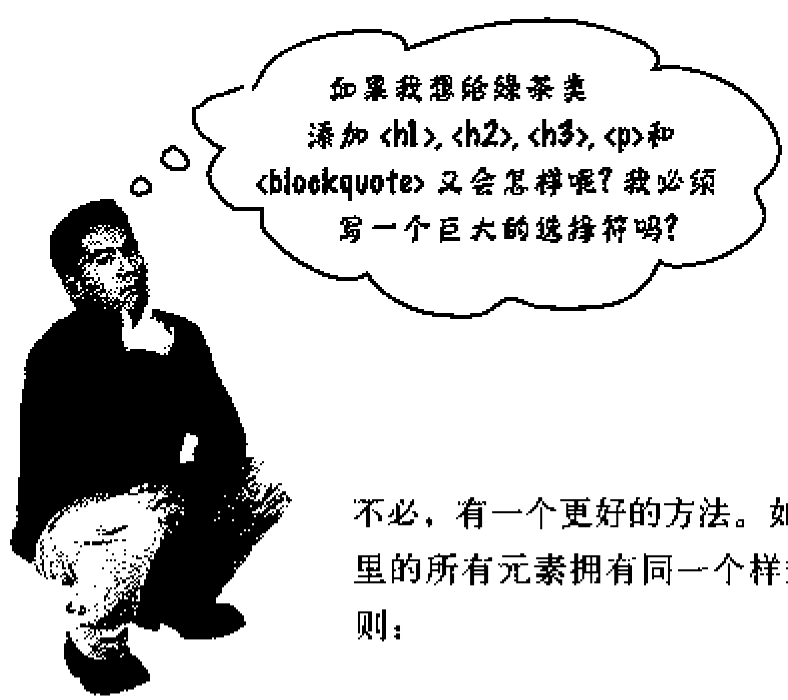
但如果你想要对所有**<blockquote>**进行相同的操作呢？那么，你可以这么做：

```
blockquote.greentea, p.greentea {
        color: green;
}
```

只需添加另一个选择符来处理绿茶类里的<blockquote>。现在这个规则将适用于绿茶类里的<p>元素和<blockquote>元素。

而在你的XHTML里，你应这么写：

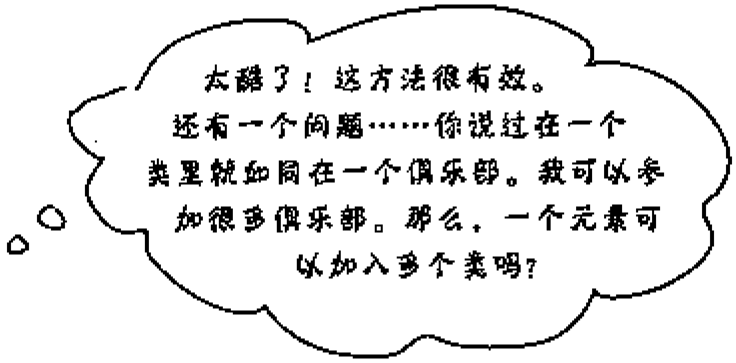
```
<blockquote class="greentea">
```



不必，有一个更好的方法。如果你想让**greentea**类里的所有元素拥有同一个样式，你可以这样编写规则：

```
.greentea {
        color: green;
}
```

如果你省略所有的元素名称，而用一个句点并且后接一个类名，那么这个规则将会适用于该类内的所有成员。



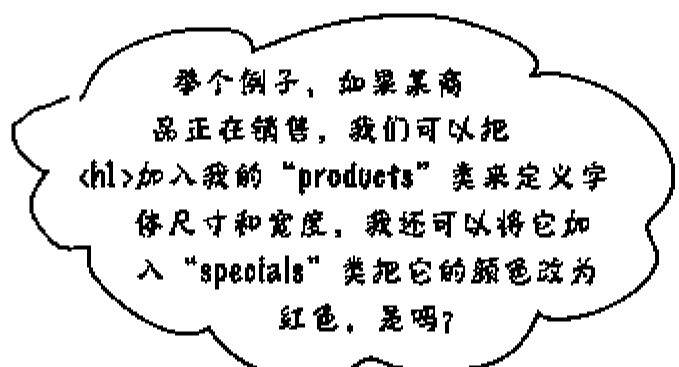


是的，一个元素可以加入多个类。

把一个元素加入多个类是很容易的。假定你要定义一个属于**greentea**、**raspberry**和**blueberry**类的**<p>**元素。在开始标记里进行如下操作：

```
<p class="greentea raspberry blueberry">
```

将每个类名赋给"class"属性值，类名之间用一个空格隔开。顺序无关紧要。





没错，如果你想让一个元素拥有不同类中定义的样式，你可以让它同时加入多个类。比如现在的情形，与商品类关联的**<h1>**元素都有一个特定的样式，但不是所有的产品都同时降价销售。在另一个类里设置了"specials"颜色，你就可以很容易地把要降价销售的产品添加到"specials"类中，使它们拥有你想要的红色。

现在，你也许想知道，如果一个元素属于多个类，而这些类定义的是相同的属性——就如上面的**<p>**元素，那结果会怎样呢？你知道的，这些类各有一个颜色属性的定义。哪一个样式会被应用呢？那么，此段落将会是绿色，蓝色，还是紫色呢？

等你掌握更多的CSS，我们会详细地对此进行讨论，而下一页中简洁的介绍会让你对它有所了解。

关于样式应用的最快捷最小巧的手册

元素、文件树、样式规则和类……这些会把你完全搞晕。如何理顺这些东西从而让你清楚哪个样式被应用于哪个元素呢？我们说过，你必须掌握更多的CSS，才能完整地回答这个问题，并且在下面几章你将会学到。但在此之前，让我们先粗略地了解一下关于如何应用样式的大致原则。

首先，是否有选择符选择你的元素？

假设你想知道一个元素的**font-family**属性值，你首先要核实的是：在你的CSS文件里是否有选择符选择你的元素？如果有，确定它有个**font-family**属性和值，而这值就是用来修饰你的元素的。

那么继承呢？

如果没有选择符与你的元素匹配，那你就得依赖继承了。观察该元素的父母，父母的父母，依次上推，直到你找到被定义的属性。如果你找到了，那就是你要找的值。

还是没找到？那么使用默认值。

如果你的元素没有从它的任何祖先那里继承属性值，那么，你将使用浏览器定义的默认值。事实上，这种情况比我们现在描述得要复杂，稍后我们将在书中介绍一些细节。

如果有多个选择符选择一个元素呢？

这是我们让一个段落属于三个类的情形：

```
<p class="greentea raspberry blueberry">
```

有三个选择符匹配这个元素并且都定义了颜色属性。我们称之为“冲突”。哪个规则能胜出呢？如果一个规则比其他的更具体，那它将胜出。但更具体是什么意思？我们将在随后的章节详细介绍如何确定一个选择符的具体性，但是现在先让我们看一些规则并进行思考：

```
p { color: black;}
.greentea { color: green; }
p.greentea { color: green; }
p.raspberry { color: blue; }
p.blueberry { color: purple; }
```

这是选择所有段落元素的规则。

这条规则选择了绿茶类的所有成员。此规则比前一条稍微具体一点。

这条规则只选择绿茶类里的段落，比上一条更具体。

这些规则也只选择某个特定类里的段落。因此它们的具体性与p.greentea规则相同。

是否还没有一个明显的胜出者？

如果你让一个元素只属于**greentea**类，那么将有一个明显的胜者：**p.greentea**选择符是最具体的，因此文本将会是绿色的。但如果你让一个元素属于所有的三个类：**greentea**、**raspberry**和**blueberry**。那么**p.greentea**、**p.raspberry**和**p.blueberry**都会选择这个元素，并且它们具有相同的具体性。现在你该怎么办呢？选择在CSS文件里最靠后的规则。如果你因为两个选择符具有相同的具体性而无法解决冲突，可以使用你样式表里的规则排序来解决冲突。也就是，你使用CSS文件里最靠后（最接近底部）的规则。而在上面的例子中，我们将选择**p.blueberry**规则。



练习

在你的“lounge.html”文件里，把绿茶段落改为包含所有类，如下：

```
<p class="greentea raspberry blueberry">
```

保存，然后重新载入。现在关于酷爽绿茶的那个段落是什么颜色？ ________

下一步，在你的XHTML里重新排列类：

```
<p class="raspberry blueberry greentea">
```

保存，然后重新载入。现在酷爽绿茶段落是什么颜色？ ________

下一步，打开你的CSS文件并把p.greentea规则移到文件的底部。

保存，然后重新载入。现在酷爽绿茶段落是什么颜色？ ________

最后，把p.raspberry移到文件的底部。

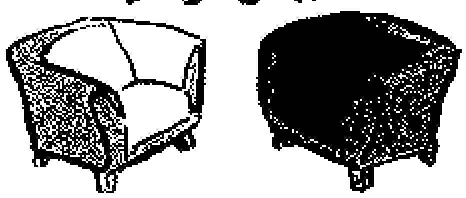
保存，然后重新载入。现在酷爽绿茶段落是什么颜色？ ________

完成后，重新编写绿茶元素使它看起来与最初的一样：

```
<p class="greentea">
```

保存，然后重新载入。现在酷爽绿茶段落是什么颜色？ ________

围炉夜话



今夜话题：CSS和XHTML语言比较

CSS： 你看到了吗？我就像Houdini！我打破你的**<style>**元素并在我的文件里重组。如你在第1章说的，我从没有逃避责任。

XHTML： 别激动，我依然不得不链接你，让你完全有用。

CSS： 不得不链接我？你知道没有我的样式你的网页就无法合理布局。

XHTML： 让我们回顾一下……当我和我的元素在努力使网页保持结构化的时候，你却正在研究如何让头发更亮和怎么给指甲染色呢。

CSS： 如果你对本章多加留意，你会发现我所做的工作是十分强大的。

XHTML： 好，我同意这点；使用CSS确实使我的工作变得更容易了。在我眼里，所有那些旧的、已被淘汰的样式元素都是麻烦。我确实喜欢无需在XHTML里插入一串资料就可以修饰我的元素，只是有时加入一个类属性。

CSS： 现在，那不是更好吗？我喜欢新的属性。

XHTML： 但我仍然没有忘记你是如何调侃我的语法结构的……<remember>?

CSS

你不得不承认XHTML有点笨拙，毕竟你应用的知识是上世纪90年代的技术。

XHTML

但是它经历了时间的考验。你认为CSS是一流的？我觉得，你只不过是一些规则，哪算得上是一门语言呢？

CSS

你在开玩笑吗？我是非常有表现力的。我可以选择我要选择的元素，并把它描述为我想要的样式。你只是刚开始见识我强大的表现力而已。

XHTML

哦，是吗？

CSS

当然，等着瞧吧。我能把字体和文本设置为各种有趣的样式。我甚至能控制各个元素如何支配它们周围的空间。

XHTML

嗯，听起来好像你的功能很强大，我不太喜欢那说法。毕竟，我的元素希望能够控制自己。

CSS

哈哈。你认为用`<style>`标记就能控制我了。你将会看到，只要我愿意，我可以让你的元素坐下、大叫甚至打滚儿。

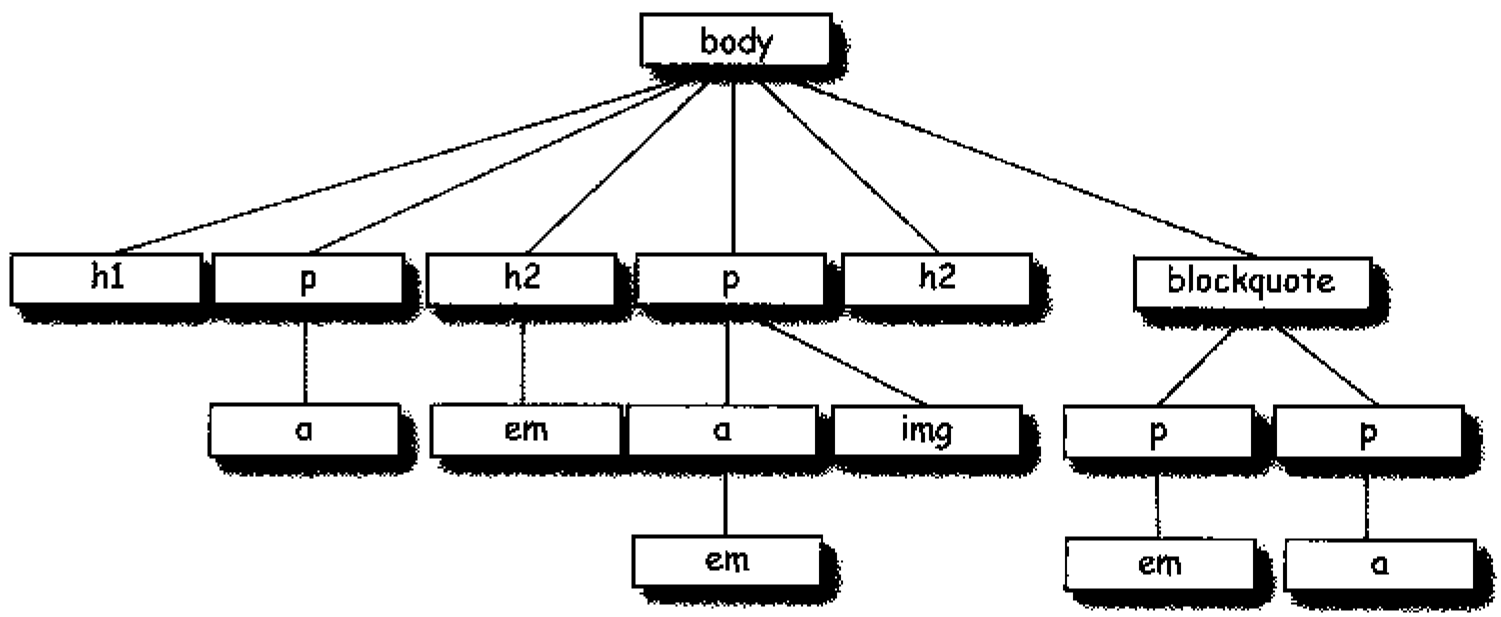
XHTML

停！停！保安……保安？！



谁继承了属性？

呼……<body>元素到天上去找伟大的浏览器去了,但它留下了许多子孙和一个“绿”颜色的继承属性。你将会在下面看到它的家族树。标记所有继承了<body>元素的绿颜色的子孙。在此之前别忘了先看看下面的CSS。




```
body {
        color: green;
}
p {
        color: black;
}
```

这是CSS。用这个来决定上面哪个元素中要变成绿色。

扮演浏览器

下面，你将会发现CSS文件“style.css”中有一些错误。你的任务是扮演浏览器找出所有的错误。做完这个练习后，你可以对照本章末尾的答案，看看你是否找出了全部的错误。

如果你的CSS有错误，通常错误后面的所有规则都会被忽略。因此，要通过做这个练习，养成检错的习惯。



“style.css”文件

```
<style>

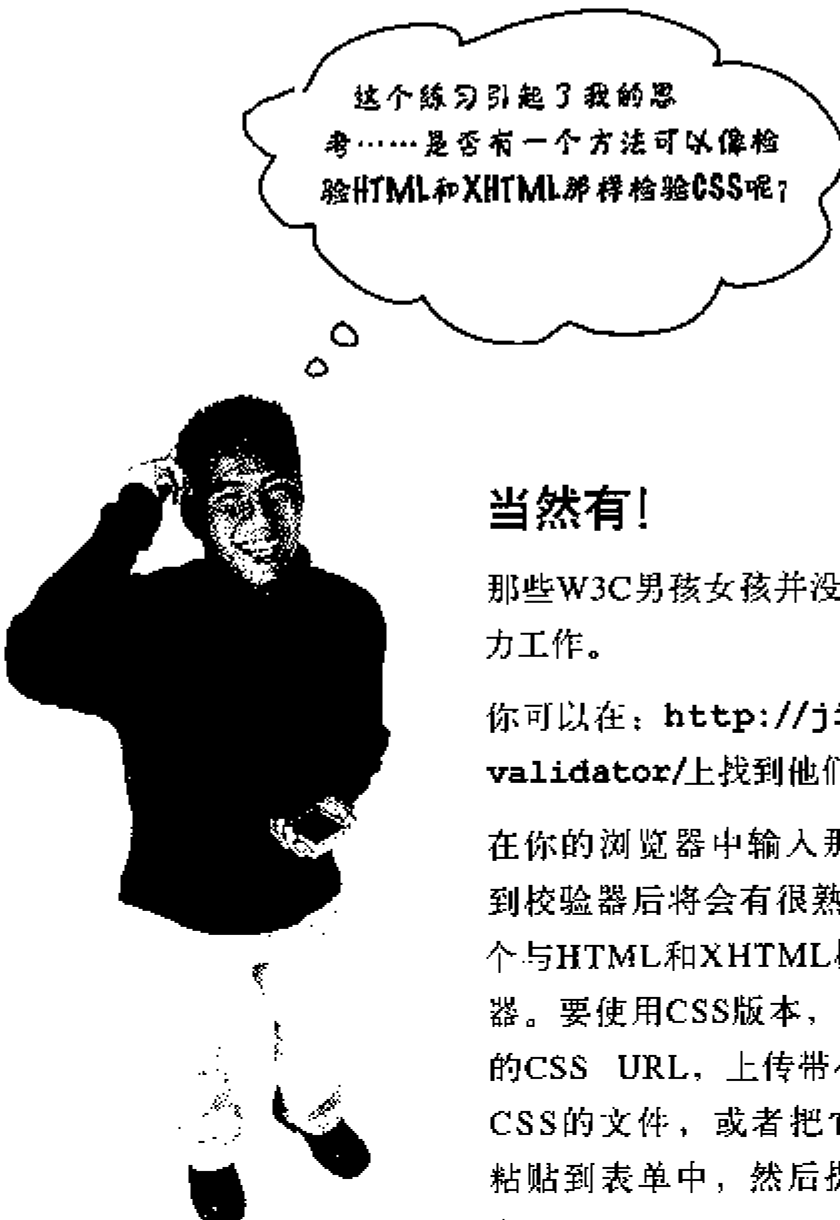
body {
    background-color: white

h1, {
    gray;
    font-family: sans-serif;
}

h2, p {
    color:
}

<em> {
    font-style: italic;
}

</style>
```



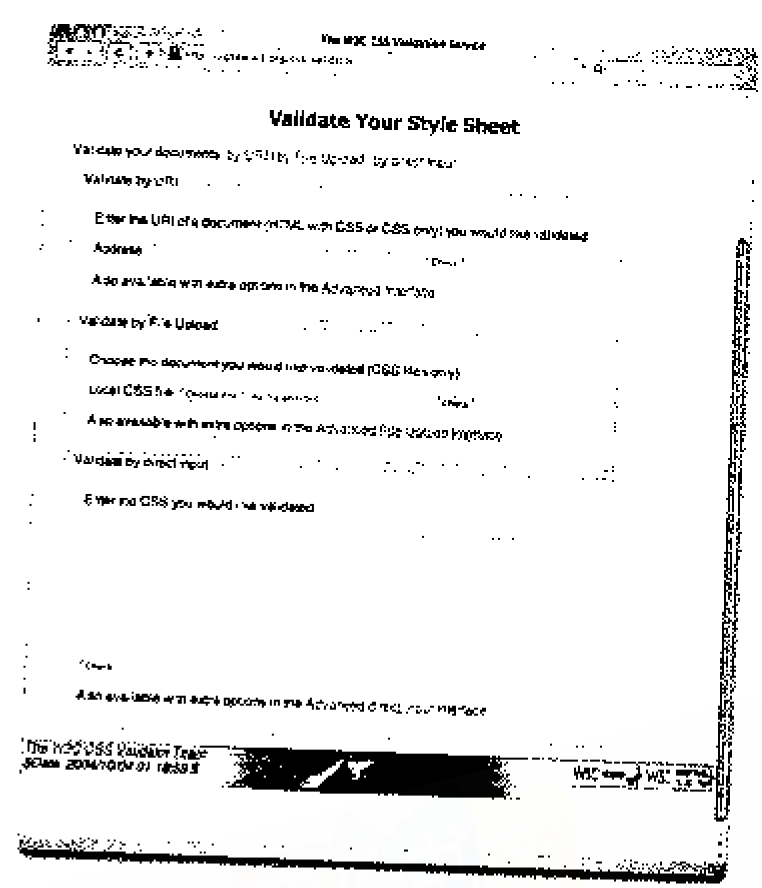
当然有！

那些W3C男孩女孩并没有闲着，他们一直在努力工作。

你可以在：**http://jigsaw.w3.org/css-validator/**上找到他们的CSS校验器。

在你的浏览器中输入那个URL，我想，你看到校验器后将会有很熟悉的感觉。你会发现一个与HTML和XHTML校验器极为相似的校验器。要使用CSS版本，你只需向校验器指明你的CSS URL，上传带有CSS的文件，或者把它粘贴到表单中，然后提交。

毫不奇怪你会需要DOCTYPE或CSS字符编码。继续进行，马上试一下（要知道下一页可能就不会给你机会了哦）。



确保休闲室的CSS合法

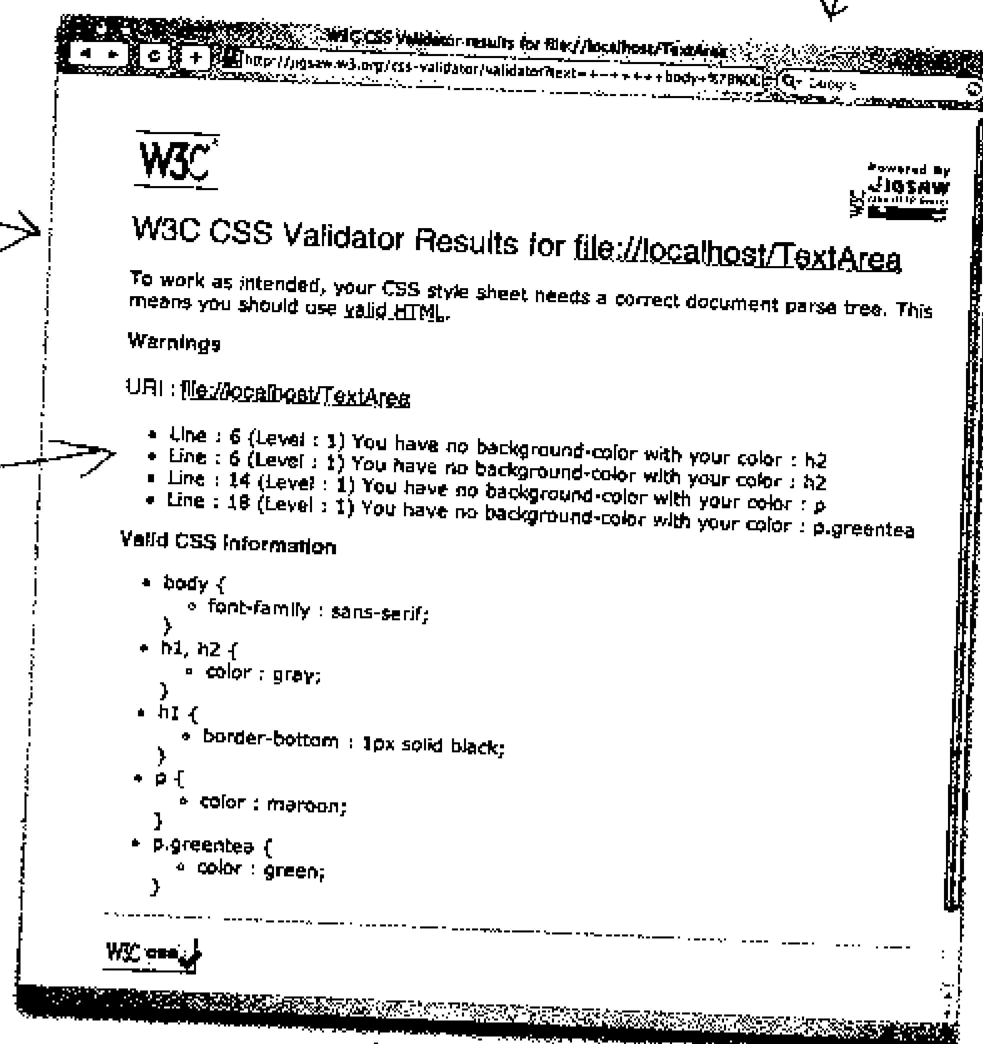
在你总结本章内容之前，你是否会觉得，如果所有的Head First休闲室CSS都经过验证那将会更好？是的，你会。用你喜欢的方法把你的CSS提交给W3C。如果服务器上有你的CSS文件，就把CSS URL输入表单中；否则要么上传你的CSS文件，要么复制粘贴CSS到表单中（如果你上传文件，确保表单指向的是你的CSS文件，而不是XHTML文件）。做完这个后，单击“Check”按钮。

如果你的CSS不合法，用前几页的CSS核对它并找出你犯的任何细微的错误，然后重新提交它。

这提示你CSS需要与正确的XHTML结合，因此，要确保你的XHTML（或者HTML）也是合法的。

这是关于CSS的一些警告。与其说这些是警告，还不如说是建议。举个例子，这些警告都是用来告诉你给标题和段落设置背景颜色的。

这些是合法的CSS，而这正好是你全部的CSS，也就是说，你的CSS是合法的。



与你验证XHTML时不同，你通过验证的时候没有“绿色的成功标志”。因此要检查网页顶端看看是否有“Errors”。如果你没有看到，就表明你的CSS通过验证了。

there are no Dumb Questions

问： **我必须在意那些警告吗？或者按他们所说的去做吗？**

答： 最好是查看一下这些警告，但你会发现其中一些是属于建议的范畴，而非“必做的”范畴。只要有一些差错，验证器就会将之判为错误，因此，还是应该正视这些警告。

Property Soup

top
控制元素顶部的位置。

text-align
用这个属性让文本左对齐、右对齐或居中。

letter-spacing
这个属性用来设置字母之间的间距，像 这 样。

color
用来设置文本元素的字体颜色。

background-color
这个属性用来控制元素的背景颜色。

font-style
用这个属性得到斜体文本。

font-weight
这个属性用来控制文本的粗细，用它把文字加粗。

border
这个属性用来在元素周围加边框，可以设置成实心边框、虚线边框……

list-style
这个属性用来设置列表项的样式。

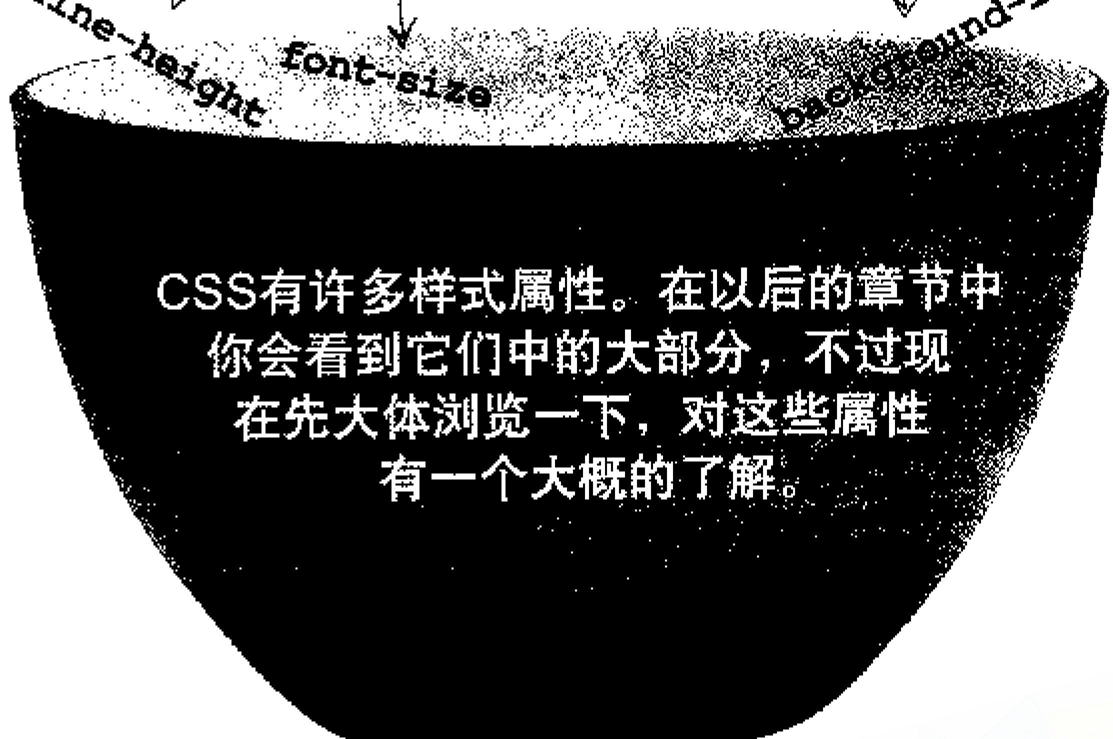
left
用它告诉元素如何放置它的左侧。

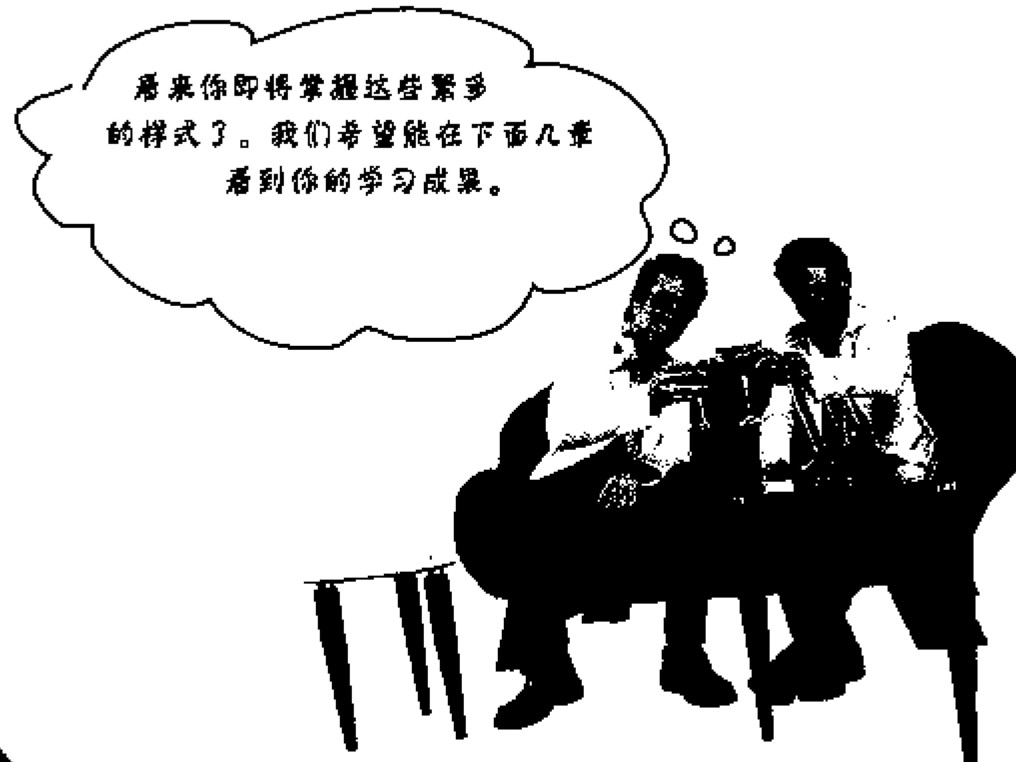
margin
如果想在元素的边缘和内容之间有一些空间，就用边界。

line-height
这个属性用来设置文本元素的行间距。

font-size
把文字变大或变小。

background-image
用这个属性把一个图像放到元素后面。



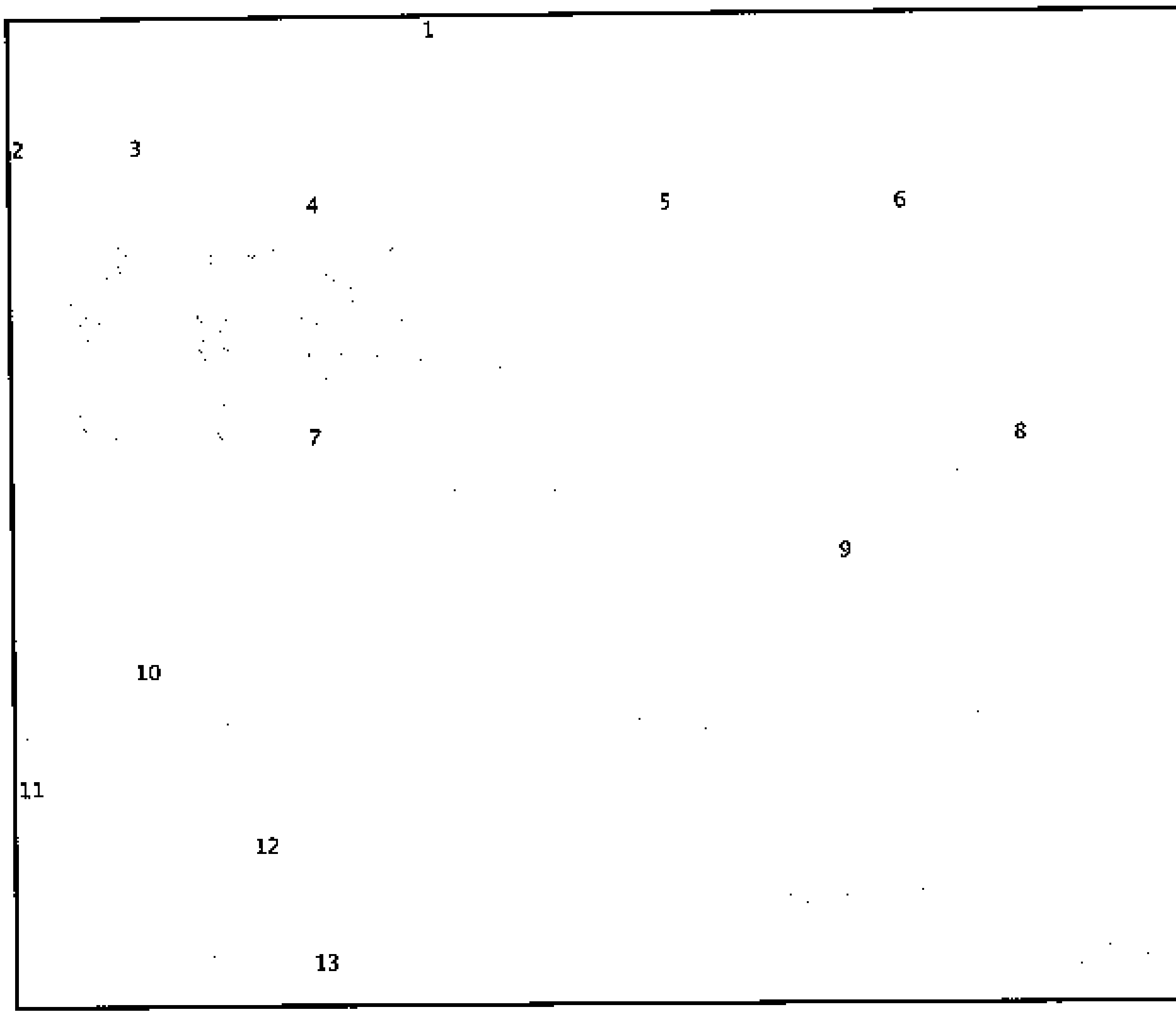


要点

- CSS中简单的表达式，称为规则。
- 每个规则为选定的XHTML元素提供样式。
- 一个典型的规则包括一个选择符、若干属性和属性值。
- 选择符指定对哪些元素应用规则。
- 每个属性声明以一个分号结束。
- 一个规则里的所有属性和属性值包含在括号{}之间。
- 你可以用元素名作为选择符来选择任意元素。
- 你可以一次选择多个元素，只需用逗号把那些元素名隔开就可以了。
- 在HTML中引入一个样式的最简单的方法是用<style>标记。
- 你应当给XHTML和复杂的站点链接一个外部样式表。
- <link>元素用来引入一个外部样式表。
- 许多属性可以被继承。举个例子，如果给<body>元素设置一个可被继承的属性,那么<body>的所有子孙元素将继承这个属性。
- 你可以给你要修改的元素创建一个更具体的规则,来覆盖继承的属性。
- 用**class**属性将元素加入到一个类中。
- 在元素名和类名之间加“.”来选择类里的某个元素。
- 用“.classname”来选择类里的所有元素。
- 一个元素可以属于多个类,只需在class属性里放置许多类名,并用空格把它们隔开。
- 在http://jigsaw.w3.org/css-validator上,你可以用W3C验证器验证你的CSS。

XHTML填字游戏

休息一下，来做游戏吧，这可是能帮你掌握CSS哦！

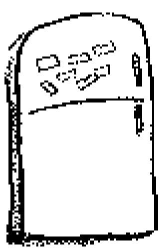


横排提示：

1. 定义一组元素。
2. 某些字体的装饰部分。
4. 样式在这里面被定义。
7. 无衬线字体。
9. 每个规则定义了一系列属性和_____。
10. 元素如何从它父母那里获取属性。
11. 用这个元素来引入一个外部样式表。
12. 选择元素。
13. 真实显示。

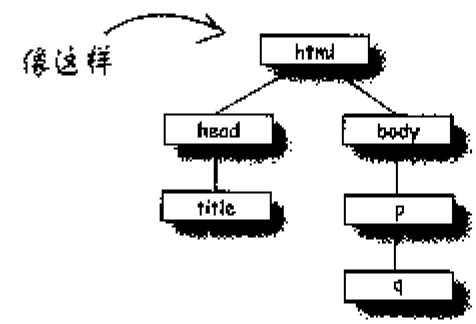
竖排提示：

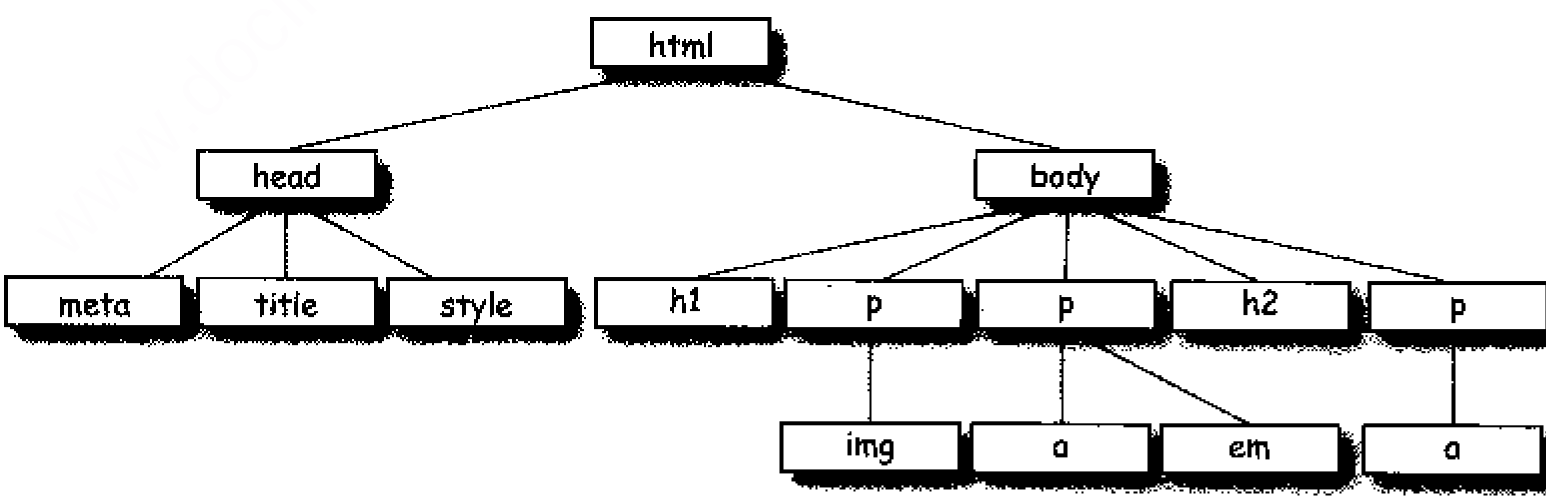
1. 通过继承,设置给一个元素的属性也可以传递给它的_____。
2. 在HTML文件里，你可以把你的CSS置于这些标记里面。
3. 这次他们成功了,因为他们使用了外部样式表。
5. 设置字体颜色的属性。
6. 设置字体类型的属性。
7. 外部样式文件的称谓。
8. 他们确实需要一些样式。



标记帖答案

记得我们在第3章画的XHTML元素图吗?你为休闲室的主页重做了一次。这是答案:

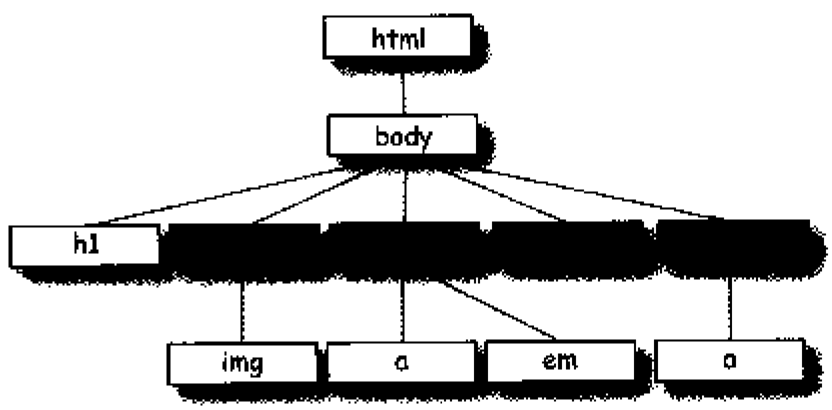




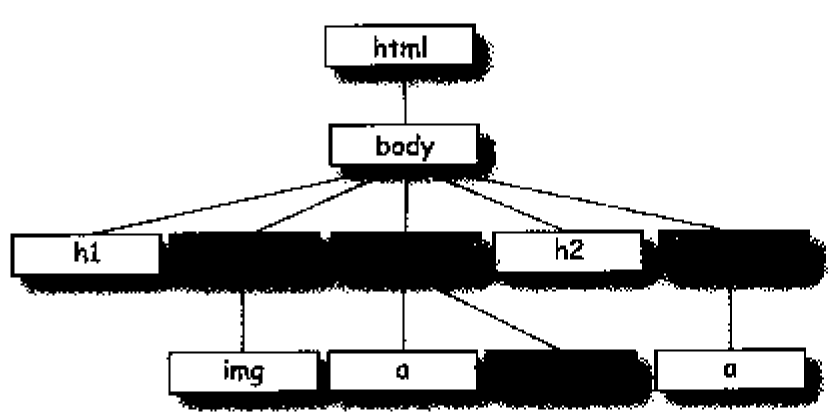
Sharpen your pencil 答案

被选择的元素都着色了。

```
p, h2 {
    font-family: sans-serif;
}
```




```
p, em {
    font-family: sans-serif;
}
```



Sharpen your pencil

答案

轮到你了：添加两个类——“raspberry”和“blueberry”到“elixir.html”里的相应段落中，然后编写样式分别将文本设置为蓝色和紫色。raspberry属性的值设为“blue”，而blueberry的属性值设为“purple”。

```
body {
        font-family: sans-serif;
}

h1, h2 {
        color: gray;
}

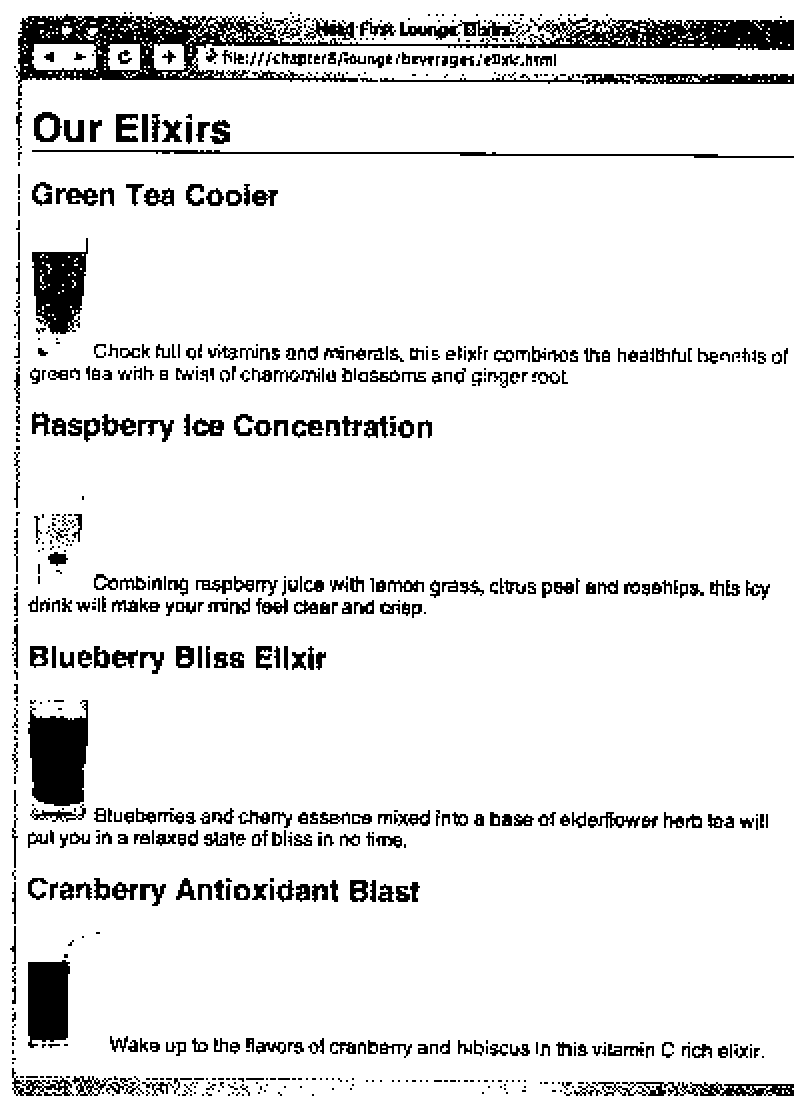
h1 {
        border-bottom: 1px solid black;
}

p {
        color: maroon;
}

p.greentea {
        color: green;
}

p.raspberry {
        color: blue;
}

p.blueberry {
        color: purple;
}
```



Sharpen your pencil 答案

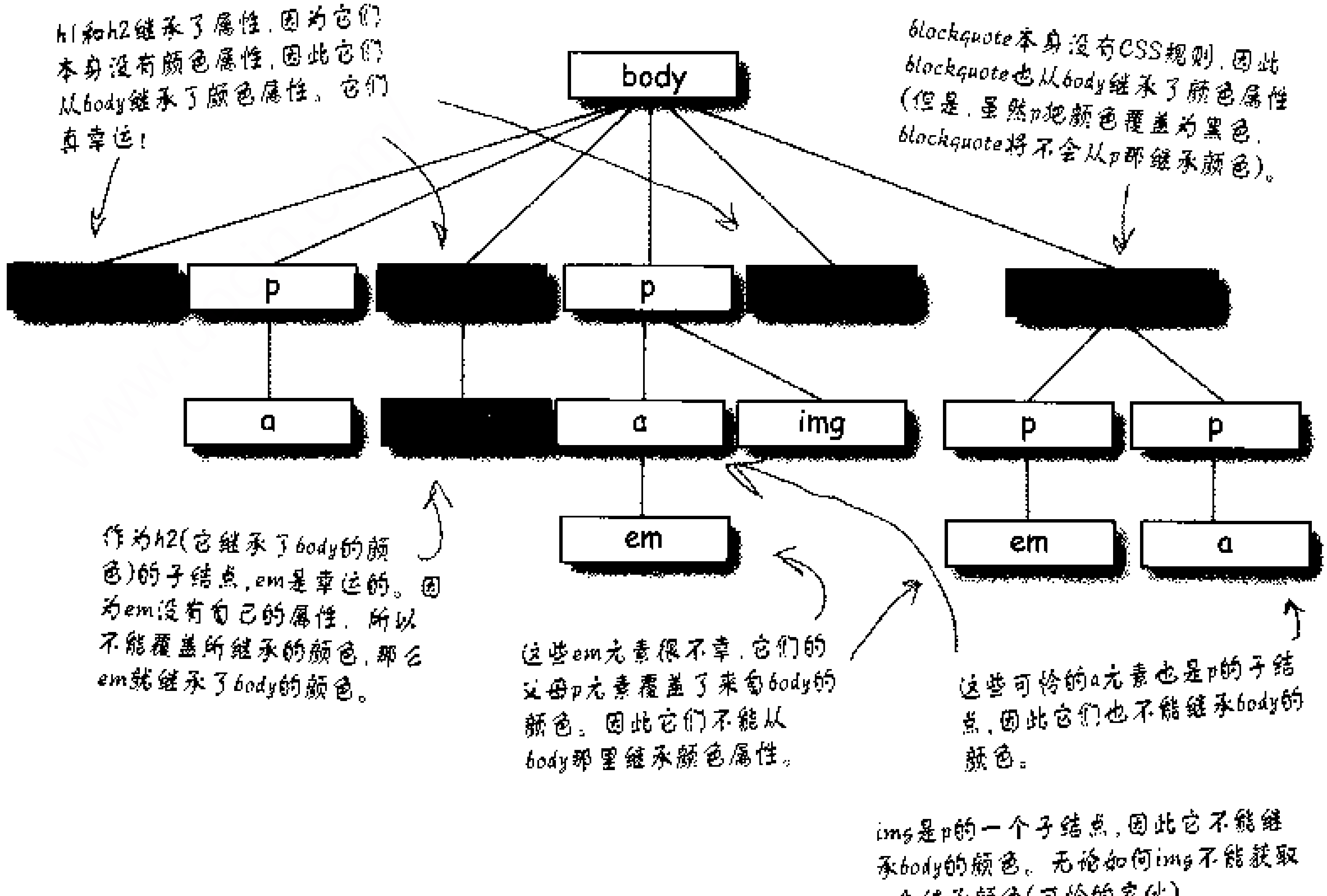
```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
    <title>Head First Lounge Elixirs</title>
    <link type="text/css" rel="stylesheet" href="../lounge.css" />
  </head>
  <body>
    <h1>Our Elixirs</h1>
    <h2>Green Tea Cooler</h2>
    <p class="greentea">
          <img src="../images/green.jpg" />
          Chock full of vitamins and minerals, this elixir
          combines the healthful benefits of green tea with
          a twist of chamomile blossoms and ginger root.
    </p>
    <h2>Raspberry Ice Concentration</h2>
    <p class="raspberry" >
          <img src="../images/lightblue.jpg" />
          Combining raspberry juice with lemon grass,
          citrus peel and rosehips, this icy drink
          will make your mind feel clear and crisp.
    </p>
    <h2>Blueberry Bliss Elixir</h2>
    <p class="blueberry" >
          <img src="../images/blue.jpg" />
          Blueberries and cherry essence mixed into a base
          of elderflower herb tea will put you in a relaxed
          state of bliss in no time.
    </p>
    <h2>Cranberry Antioxidant Blast</h2>
    <p>
          <img src="../images/red.jpg" />
          Wake up to the flavors of cranberry and hibiscus
          in this vitamin C rich elixir.
    </p>
  </body>
</html>
```

习题解答

谁继承了属性？

```
body {
    color: green;
}

p {
    color: black;
}
```





习题解答



扮演浏览器

下面，你将会发现CSS文件“style.css”中有一些错误。你的任务是扮演浏览器并找出所有的错误。你找到全部错误了吗？

```
<style>

body {
    background-color: white

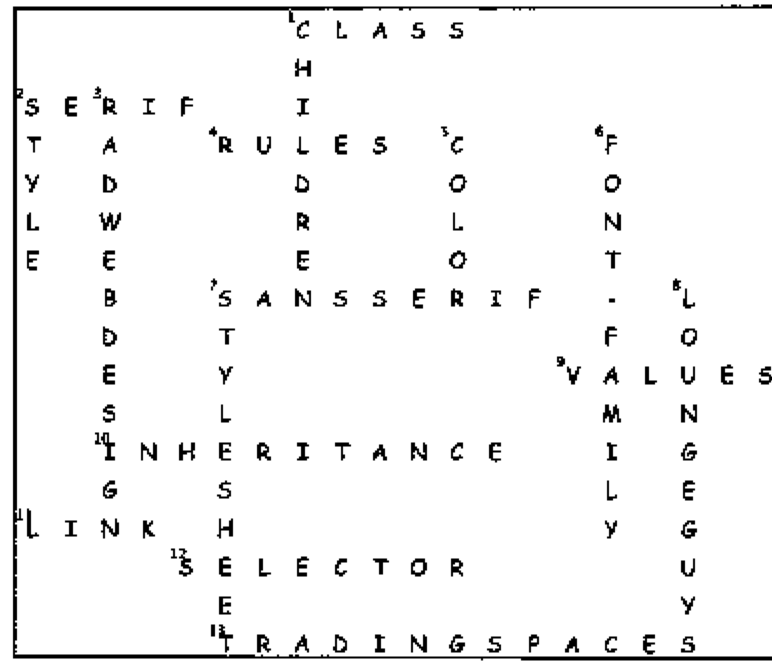
h1, {
    gray;
    font-family: sans-serif;
}

h2, p {
    color:
}

<em> {
    font-style: italic;
}

</style>
```

- `<style>`：在你的CSS里不该有XHTML！<style>标记是XHTML，并且它在CSS样式表里无法运行。
- `background-color: white`：缺少分号。
- `body {`：缺}。
- `h1,`：多余的逗号。
- `gray;`：缺少属性名。
- `color:`：缺少属性值和分号。
- `<em>`：把元素名错用为XHTML标记。应该是em。
- `</style>`：在CSS里不需要</style>标记。



练习答案

在你的“lounge.html”文件里，把绿茶段落改为包含所有的类，如下：

`<p class="greentea raspberry blueberry">`

保存，然后重新载入。现在关于酷爽绿茶的那个段落是什么颜色？ **紫色**

<p>元素是紫色，因为在CSS文件里，blueberry规则最靠后。

下一步，在你的XHTML里重新排列类：

`<p class="raspberry blueberry greentea">`

保存，然后重新载入。现在酷爽绿茶段落是什么颜色？ **紫色**

<p>元素仍然是紫色，因为class属性里的类名的顺序与类的选择没有关系。

下一步，打开你的CSS文件并把p.greentea规则移到文件的底部。

保存，然后重新载入。现在酷爽绿茶段落是什么颜色？ **绿色**

现在，<p>元素是绿色，因为在CSS文件里，greentea规则最靠后。

最后，把p.raspberry移到文件的底部。

保存，然后重新载入。现在酷爽绿茶段落是什么颜色？ **蓝色**

现在，<p>元素是蓝色，因为在CSS文件里，raspberry规则最靠后了。

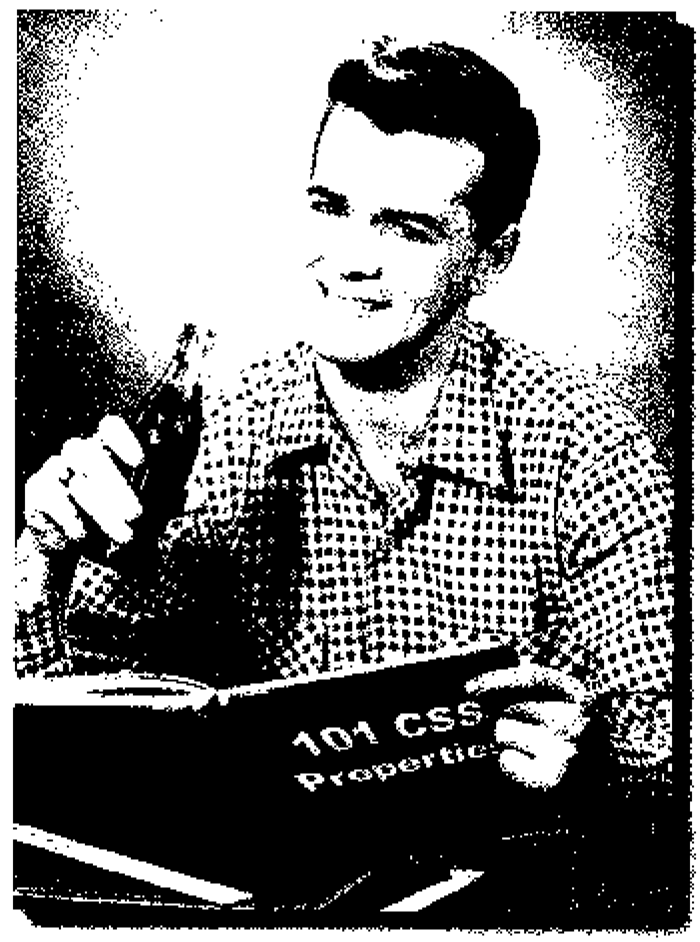
完成后，重新编写绿茶元素使它看起来与最初的一样：

`<p class="greentea">`

保存，然后重新载入。现在酷爽绿茶段落是什么颜色？ **绿色**

好了，现在<p>元素只属于一个类，因此，我们采用最具体的规则——p.greentea。

扩大你的词汇量



愉快的CSS语言课程就要开始了。 你已经了解了CSS的基础，知道如何创建CSS规则来选择和定义一个元素的样式。现在该扩大你的词汇量了，这意味着你要学习一些新的属性及它们的功能。在这章中我们将用那些能影响文本显示的、最普通的属性来完成工作。为此，你需要学习一些关于字体和颜色的知识。你将会发现你不必始终只用大家都用的字体，或浏览器默认的段落和标题中粗笨的字体大小和样式。你会发现更多新奇的颜色。

从一定的高度看文本和字体

CSS有许多样式化文本的属性。本章将主要介绍如何使用CSS来控制文本的字体、格式、颜色和文本修饰等。我们将从显示页面的字体开始学起，你已经见过**font-family**属性了，本章中将开始学习特定字体的更多知识。

深入学习CSS之前，先看看用来指定和改变字体外观的属性。之后，我们将逐一学习各种字体及其详细的用法。

用font-family属性定义网页中的字体。

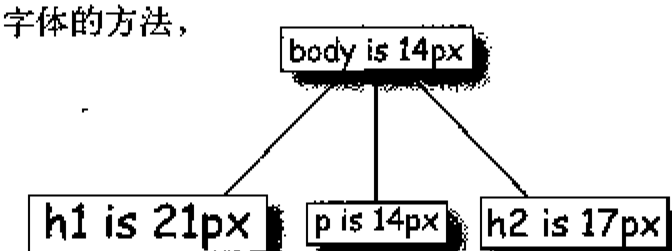
字体对网页的设计有很大的影响。在CSS中，字体被分成几个“字体系列”。你可以从中选择自己喜欢用在网页设计中的字体系列来设定字体。大多数电脑一般只安装了特定的几种字体，所以选择时要慎重。在这一章中，你将学习到所有用来指定和充分利用字体的知识。

```
body {
      font-family: Verdana, Geneva, Arial, sans-serif;
}
```

Andale Mono
Arial
Arial Black
Comic Sans
Courier New
Georgia
Impact
Times New Roman
Trebuchet MS
Verdana

用font-size属性控制字体大小。

字体大小也对网页的设计和可读性有很大的影响。用CSS定义字体大小的方法有很多，在这章中我们将一一介绍。你还将学到一种指定字体的方法，它允许用户改变字体大小而不影响你的设计。

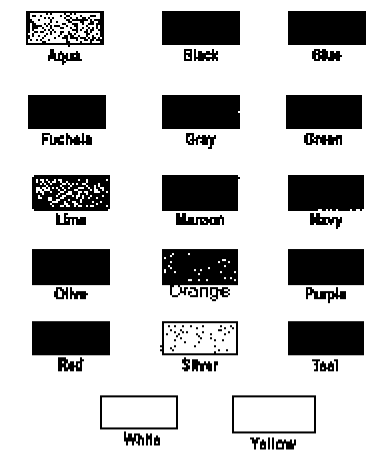



```
body {
      font-size: 14px;
}
```

用color属性给文本添加颜色。

可以用color属性改变文本的颜色。为此，你需要了解一些关于网页颜色的知识。我们会全面介绍颜色，其中包括神秘的“十六进制代码(hex codes)”取色。

```
body {
        color: silver;
}
```



用font-weight属性设置字体的粗细。

你可以把字体设置成不同的粗细，为什么还要用那些单一而又最普通不过的字体呢？或者是字体太粗了，要变成普通粗细？所有这些使用**font-weight**属性就能轻松解决。

```
body {
        font-weight: bold;
}
```

lighter

normal

bold

bolder

用text-decoration属性给文本添加更多样式。

应用**text-decoration**属性，你可以用overlines、underlines和line-throughs等来修饰文本。20世纪90年代的Web不支持文字闪烁，但CSS的设计者甚至已经为**text-decoration**增加了blink值来修饰文本（尽管不需要用浏览器来显示）。

```
body {
        text-decoration: underline;
}
```

none

underline

overline

~~line-through~~

blink

到底什么是字体系列呢？

你已经了解**font-family**属性了，那么你应该知道“sans-serif”这个属性值。你可以用**font-family**属性得到比这更新颖的字体，不过首先你要知道字体系列的具体内容。赶快来了解一下……

每个font-family包括一系列具有相同特征的字体。字体系列一共有五种：sans-serif、serif、monospace、cursive和fantasy。每个系列又包括许多字体，所以在这里每个系列我们只举很少的几个例子。

Sans-serif Family

Verdana　Arial Black

Trebuchet MS　Arial

Geneva

Serif 系列是有衬线（Serifs）的字体，一般用于报纸印刷。

衬线（Serifs）是字母主线端的短细线。

Sans-serif 系列是没有衬线（Serifs）的字体。在电脑屏幕上这些字体要比Serif字体更具可读性。

Sans-serif的意思是“没有衬线”。

Serif Family

Times

Times New Roman

Georgia

不同的电脑，其字体会有所不同。实际上，操作系统和用户所装的应用程序不同，可用的字体也会有所不同。所以，谨记，你的用户电脑中的字体也许跟你电脑中的字体不一样。

Monospace Family

Courier

Courier New

Andale Mono

Monospace 系列的字母具有相同宽度。例如，字母"i"与字母"m"所占的水平空间是相同的。这些字体主要用来显示软件代码实例。

Cursive 系列是一些看起来像是手写的字体，有时用于标题。

仔细观察字体系列：serif字体看起来古典而优美，而Sans-Serif字体看起来清晰而易读。Monospace字体像是用打字机打出来的，Cursive和Fantasy字体就比较有趣或格式化。

Cursive Family

Comic Sans

Apple Chancery

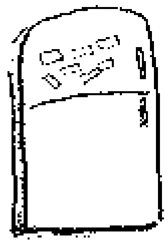
Fantasy Family

LAST NINJA

Impact

Fantasy系列是一些有固定装饰的字体，这些字体并不多见，在规范的Web设计中也不常用。

字体帖



把下面的字按字体归类，将左边方框中的单词按类别填进右边对应的字体系列下。核对答案后再开始学习下一页内容。必要的话可以复习一下前面对字体系列的描述。

Bainbridge

CARTOON

Palomino

Angel

Iceland

Messenger

Savannah

Crush

Nautica

Quarter

Monospace Family

Fantasy Family

Cursive Family

Sans-serif Family

Serif Family

用CSS定义字体系列

是的，字体系列中有许多漂亮的字体，那么我们如何把它们添加进网页呢？在前几章中你已经大体了解了**font-family**属性，也知道了如何在休闲室中把**font-family**定义成“sans-serif”。以下是一个更有趣的例子：

font-family通常指定几个不同的字体名列表，它们属于同一系列。

```
body {
    font-family: Verdana, Geneva, Arial, sans-serif;
}
```

可以用font-family定义多种字体，只要用逗号把字体名称分开就行了。

拼写出字体名，注意区分大小写。

末尾通常加一个通用的字体系列名，像“serif”、“sans-serif”、“monospace”，一会儿你就能看到它的作用。

font-family定义的工作原理

浏览器是这样解释**font-family**指定的字体的：

查找计算机中有没有Verdana这种字体，如果有，元素（在这个例子中，是<body>元素）就用这种字体。

如果没有Verdana，就查找字体Geneva，如果有，body就用这种字体。

如果连Geneva也没有，就查找字体Arial，如果有，body就用此字体。

最后，如果这些具体的字体都没有，就使用浏览器默认的某个“sans-serif”字体。

```
body {
    font-family: Verdana, Geneva, Arial, sans-serif;
}
```

不一定非得指定四种不同的字体，可以是两种、三种等。在前几章中我们只用了一种（默认的sans-serif字体），但是我们不推荐使用，因为它不能根据你自己的意愿变换字体。

用**font-family**属性可以创建自己喜欢的字体列表。但愿大多数浏览器都有你首选的一种字体，如果没有，浏览器至少可以保证能从同一系列中提供一种普通的字体。

我们来给你的网页添加一些字体……

重新看一下Tony的日志

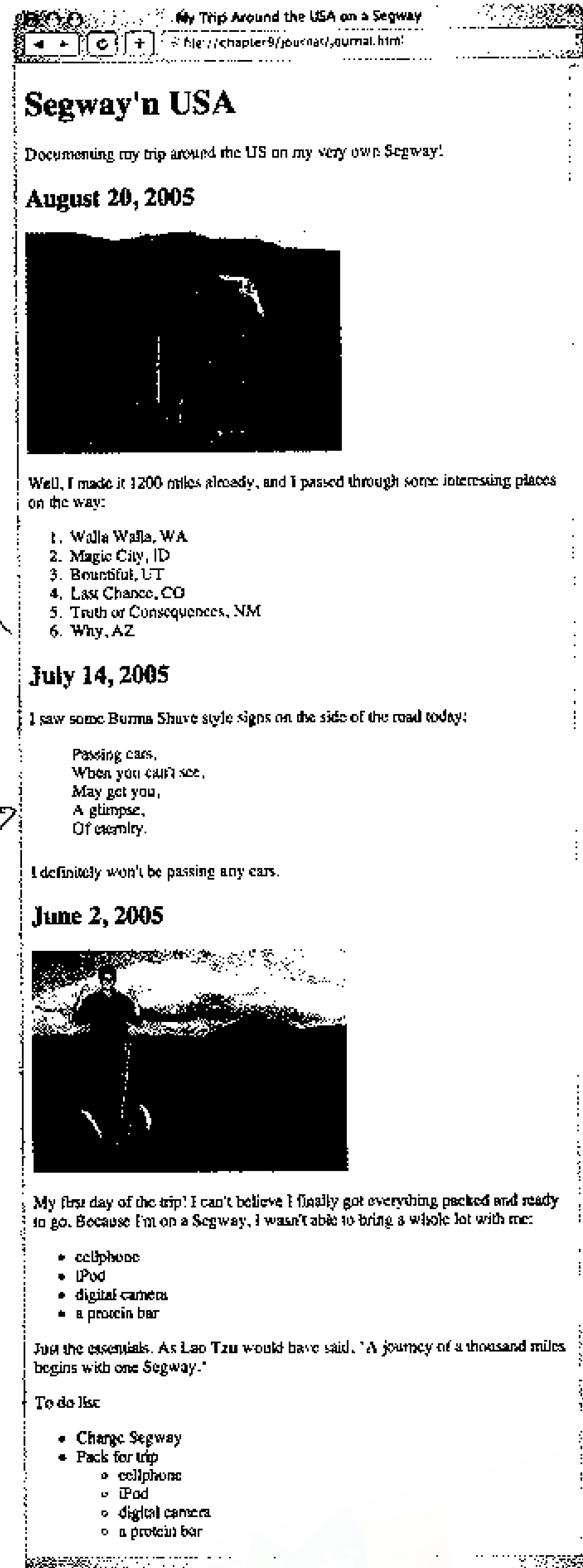
现在你知道如何定义字体了，那么我们回过头来看看Tony骑电动滑板车穿越美国的那个网页,并改变一下它的外观。我们会对Tony的网页的文本样式做一些细微的改变，增加一些东西，但不会有明显的变化。在这章结束时站点就会有一个漂亮的外观了。想想该在哪儿做一些改进，然后给Tony一个新的**`font-family`**。

我们还没有在Tony的站点中应用任何样式，所以他的站点的整个页面都只用了一种serif font-family。

默认的标题字号很大，这样不够吸引人。

引用也不够整齐，添加一些font-style就会有所改进。

这一页中除了照片，全都是一种颜色。我们还会添加一些字体颜色来使它更生动一些。



给Tony换一种新的font-family

我们来给Tony设置**font-family**。先从一些整齐的sans-serif字体开始。首先创建一个新文件，命名为“journal.css”，接着存放在“chaper9/journal”目录下，然后添加以下规则：

```
body {
    font-family: Verdana, Geneva, Arial, sans-serif;
}
```

我们在<body>元素中设置font-family属性，记住<body>中的元素都会继承这些字体。

你能够在大多数PC上看到Verdana。

大多数的Mac上是Geneva。

Arial在两种机型上都很常见。

如果前面那些都没有，就用默认的sans-serif字体。

这儿我们选了一系列sans-serif字体。

现在你需要把Tony的日志链接到新的样式表文件。打开“chapter9/journal”目录下的文件“journal.html”，添加**<link>**元素来链接“journal.css”中的样式。看看我们是怎么完成的。

还记得吧，在第7章中你把Tony的日志指定成了严格的XHTML。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
      <link type="text/css" rel="stylesheet" href="journal.css" />
      <title>My Trip Around the USA on a Segway</title>
   </head>
   <body>
    .
    .
    .
   </body>
</html>
```

在这儿链接新文件“journal.css”。

改完这些之后，保存文件，启动浏览器加载页面。

测试Tony的新字体

在浏览器中打开添加了新的CSS的网页，你会看到一组漂亮的sans-serif字体。我们来看一下这些变化……

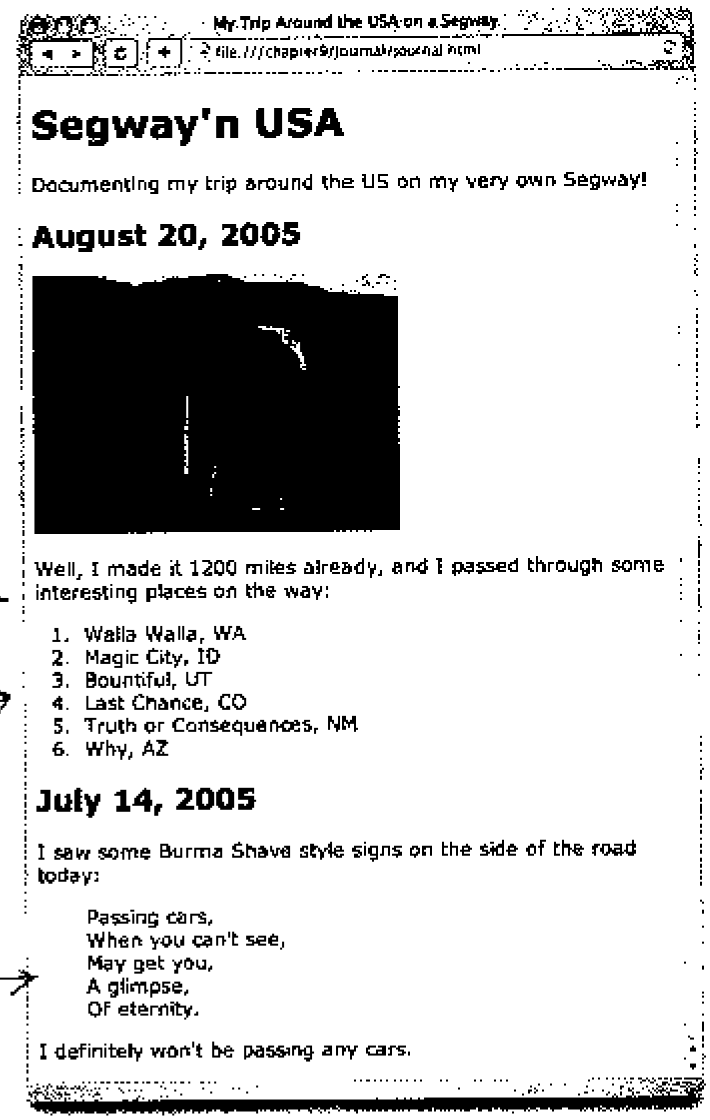
字体使Tony的网页有了全新的外观。标题字母尽管稍微有些大不过没有了衬线，看起来更清楚。

段落文本也更加清晰易读了。

由于font-family是一个可继承属性，因此网页中所有的元素都用了sans-serif字体，列表元素也不例外……

……还有<blockquote>。

如果你更喜欢serif字体，别局限于我们的例子。你可以重新声明font-family来用serif字体。



there are no Dumb Questions

问：怎么指定由几个单词组成的字体名，比如Courier New?

答： 在font-family声明中给字体名加上双引号就行了，像这样：font-family：“Courier New”，Courier。

问：font-family属性实际上是一系列不同的字体，是吗?

答： 是的，是以某种优先权排列的字体系列。排在最前面的是你最喜欢的字体，接着是喜欢的，接着是次喜欢的，依次类推。最后一种字体可以笼统地指定为通用的“sans-serif”或“serif”，只要它跟你列表中所有的字体同属一个系列就行了。

问：“sans-serif”和“serif”是真正的字体名吗?

答： 不是。不过，如果浏览器没找到前面font-family所声明的几种字体，就会找一个真正的字体来代替“sans-serif”或“serif”。而代替它的字体就是浏览器默认的那个系列中的字体。

问：我要如何选择是使用“sans-serif”还是“serif”呢?

答： 没有固定的规则。不过，在计算机的显示中，大多数人认为sans-serif是最适合body文本的。你会发现许多body文本把sans-serif或者sans-serif和serif混合使用。所以，用哪种字体完全取决于你自己的喜好，以及你想要的网页外观。

每个人都会有不同的字体，我该怎么处理？

很不幸，你不能控制用户计算机中的字体。不过，处理这个问题的最好方法是建立一个最适合于你的页面的字体列表，同时希望用户安装了其中的一种。如果没有，至少可以寄希望于浏览器提供一种同一系列中的普通字体。

至少，那是保证页面用适当字体显示的基本策略。但是由于不同的操作系统（特别是Windows和Mac）使用的字体不同，你确实需要深入学习这一点，以便做好你的工作。你需要做的是要保证**font-family**声明中包含Windows 和Mac（你的用户可能用到的别的平台，像Linux或移动设备也一样）都能出现的字体。

这儿有一个指南，它里面介绍了每个操作系统通用的一些字体。如果你需要严格控制网页中的字体，就应该多研究这方面的内容。

这些字体在Windows 和Mac中都可以使用。

Andale Mono
Arial
Arial Black
Comic Sans
Courier New
Georgia
Impact
Times New Roman
Trebuchet MS
Verdana

这些字体更可能在Mac机中找到。

Geneva
Courier
Helvetica
Times

我们再看一下Tony网页中的定义……

(1) 我们希望能用Verdana，但是……

(3) 没关系，或许我们可以指望Arial，希望它能用在Windows 或者Mac上，但如果不能……

```
font-family: Verdana, Geneva, Arial, sans-serif;
```

(2) 如果不行，Geneva也可以，而这个只能用在Mac上。但如果不能……

(4) 那也没关系，可以让浏览器选一种sans-serif字体。

调整字体大小

既然Tony有了一系列新字体，我们就需要继续工作，来调整字体大小，因为大多数人发现默认的标题字号有些大，至少从审美的角度来说是这样的。因此，你需要知道指定字体大小的方法。我们来看几种指定**font-size**的方法，然后讨论一下哪种方法最好，即能指定字体大小且便于用户使用。

如果你做得足够好，那么任何用户都可以增大网页的字号以便更容易阅读。以下两页就教你怎么做。

px

可以用像素定义字体大小，就像第5章中用于图像的像素值一样。用像素定义字体大小，就是告诉浏览器字母的高是多少像素。

```
font-size: 14px;
```

在CSS中用一个数字紧跟“px”的方式定义像素。这表明font-size应该是14像素高。

像素数字后必须紧跟px，中间不能有空格。



这是在body规则中定义font-size的方法。

```
body {
        font-size: 14px;
}
```

设置字体为14像素高意味着从字母的顶部到底部有14个像素。

%

与像素精确地规定字体大小不同，百分数用与别的字体大小的相对值来定义字体大小。

```
font-size: 150%;
```

表明字体大小应该是另一个字体大小的150%。然而，另一个字体大小指哪个父元素呢？因为**font-size**是一个从父元素继承来的属性，定义字体大小为a%，是相对于父元素的。我们看一下它是如何工作的……

这儿我们用像素定义body字体大小，用百分数150%定义一号标题字。

```
body {
        font-size: 14px;
}
h1 {
        font-size: 150%;
}
```

em

还可以用“em”定义字体大小，它跟百分数一样，是另一种相对测量单位。不过用em不是指定百分数，而是指定比例因数。以下是em的用法：

别把这跟<em>元素弄混淆了。

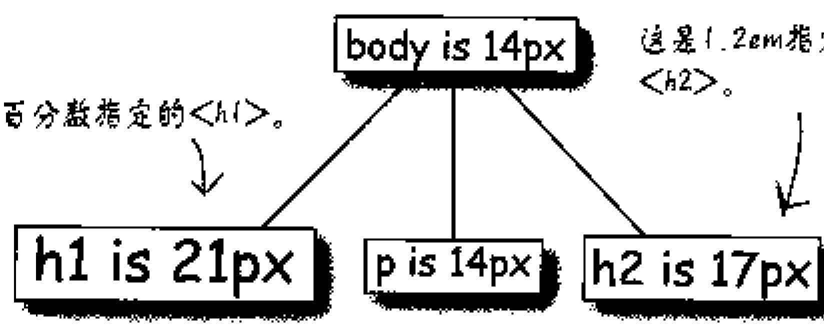
```
font-size: 1.2em;
```

这表明字体大小应该按比例放大1.2倍。

假如用这种测量法定义<h2>标题大小，<h2>标题就会是父元素字体大小的1.2倍，在这个例子中是14px的1.2倍,大约等于17px。

实际上是16.8，但大多数浏览器会四舍五入到17。

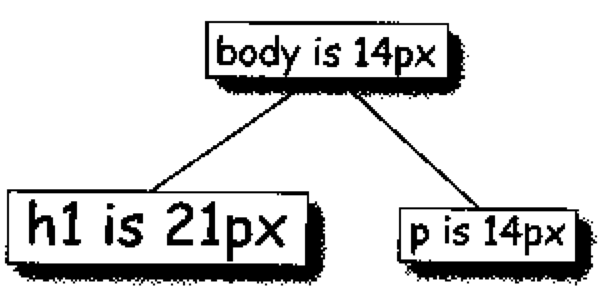
```
body {
        font-size: 14px;
}
h1 {
        font-size: 150%;
}
h2 {
        font-size: 1.2em;
}
```



这是百分数指定的<h1>。

这是1.2em指定的<h2>。

如果画一个小的文件树，你可以看到<h1>是从<body>继承来的，所以它的字体大小应该是body字体大小的150%。



标题<h1>是<body>字体大小的150%，是21 px。

我们没定义<p>的字体大小，它就继承了<body>的字体大小，是14 px。

keywords

另外还有一种定义字体大小的方法：关键字。可以把字体大小定义为：**xx-small**，**x-small**，**small**，**medium**，**large**，或者**xx-large**。浏览器将这些关键字转换成默认的像素值。

这些是各种关键字互相比较的典型大小。每种尺寸都比前一种大20%，通常把small定义为大约高12像素。需要注意的是，每个浏览器定义关键字的方式并不一定相同，如果用户需要也能自己定义。

xx-small
x-small
small
medium
large
x-large
xx-large

```
body {
      font-size: small;
}
```

在大多数浏览器中，这部分会将body文本显示为12像素。

那么，应该用哪种方法定义我的字体大小呢？

你已经学了好几种定义字体大小的方法：px,em,百分数和关键字。到底用哪一种好呢？这里有一个小技巧，可以使字体大小在大多数浏览器中显示一致。

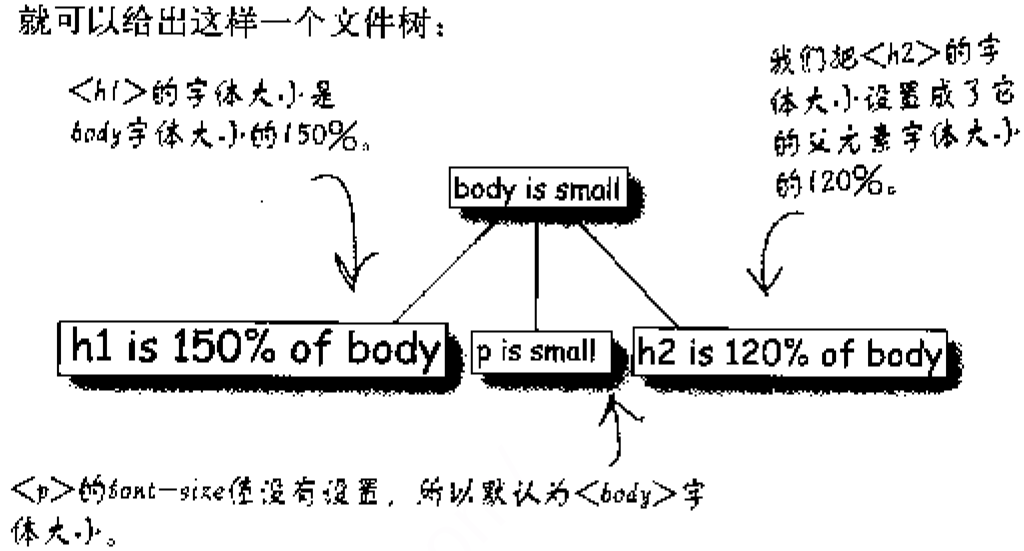
1. **选择一种关键字（推荐用small或medium）定义body字体大小，也就是网页的默认字体大小。**
2. **用em或百分数把别的元素的字体大小指定为相对于body字体大小的字体尺寸（选em还是百分数都无所谓，效果是一样的）。**

这似乎是个好技巧，但是有什么好处呢？把字体大小定义成与body字体大小相对的形式，就可以很容易地通过改变body字体大小来改变整个网页的字体大小了。想重新设计网页，把字体变大一些吗？如果你的body字体大小值是**small**，只需把它改为**medium**，所有别的元素也以相应的比例自动变大了，因为你已经把那些字体的大小定义为与body字体大小相关联的了。另外一个好处是，如果用户想调整页面中的字体尺寸，也没问题，用上这个小技巧，页面中所有的字体都会自动调整。

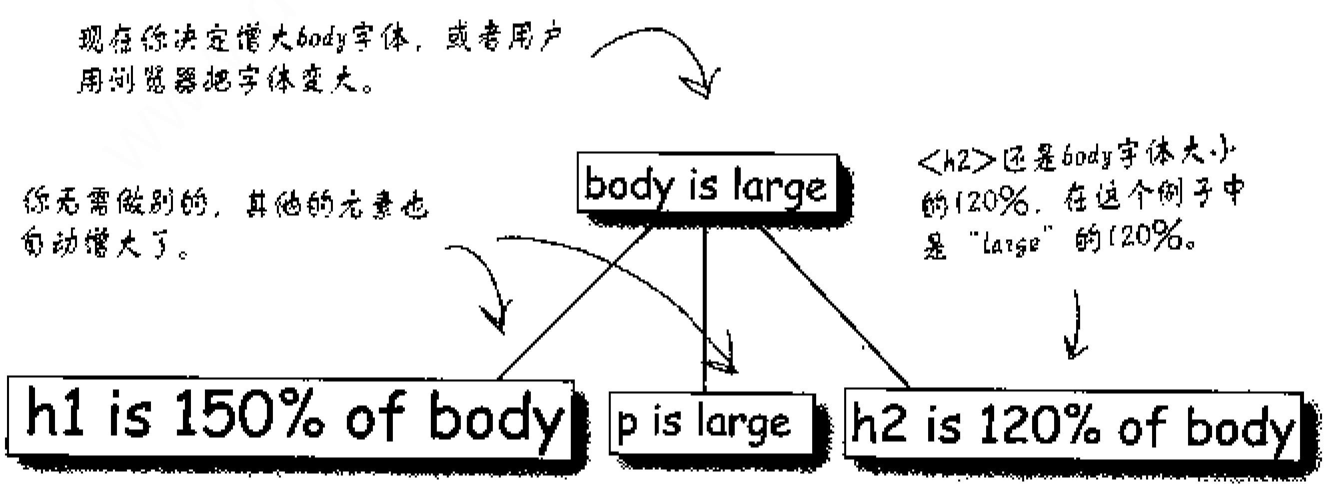
来看一下工作原理。首先，给<body>元素设置一种大小，接着把别的字体大小都设置成它的相对值，如下所示：

```
body { font-size: small; }
h1 { font-size: 150%; }
h2 { font-size: 120%; }
```

就可以给出这样一个文件树：



假如你或用户想增大页面的字体，就有一个这样的文件树：



body字体增大后，其他与之相关的字体也跟着一起增大了。这很棒，你无需再花时间改变别的字体大小；你需要做的只是改变body字体大小。所有的这一切对用户都是透明的。文本字体增大后，所有的字体都变大了，因为所有元素的字体大小都是互相关联的，所以页面在字体变大后看起来仍然不错。

注意！ **用像素指定字体大小时，IE不支持比例缩放。**

很遗憾，如果字体大小是用像素指定的，IE用户就不能调整字体大小，这也是要少用像素指定字体大小的一个原因。如果用像素，对许多用户来说就会减少页面的可用性。

幸运的是，如果你用关键字定义body字体大小，然后用em或%指定其他元素的相对大小，要求浏览器把字体变大或变小时，IE就会按比例缩放字体。

我们来改变一下Tony网页中的字体大小

应该在Tony的网页中试试这些字体大小了。把新的属性加到“chapter9/journal”文件夹下的“journal.css”文件中。改完后，把页面重新载入浏览器并观察字体大小的不同。如果没有变化，仔细检查CSS中的错误。

```
body {
    font-family: Verdana, Geneva, Arial, sans-serif;
    font-size: small;
}
h1 {
    font-size: 170%;
}
h2 {
    font-size: 130%;
}
```

根据我们的小技巧把<body>元素的font-size定义为small，其他的字体大小都将以它为基准而变化。

把别的字体大小设置成与body字体大小相关联，把<h1>设置成body字体大小的170%。

把<h2>字体大小设置得比<h1>稍小一些，即body字体大小的130%。

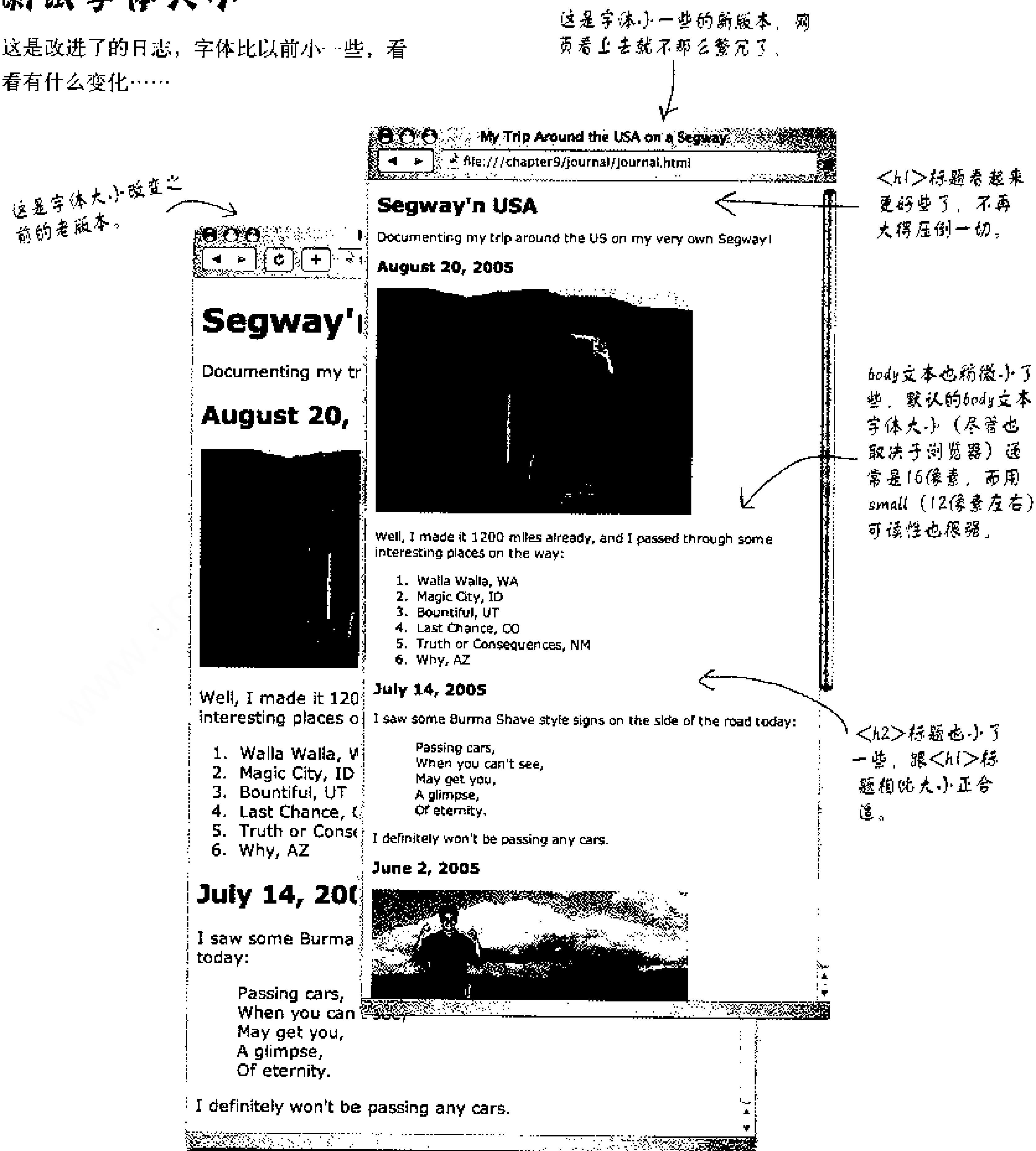
Sharpen your pencil

如果用em而不用百分数定义<h1>和<h2>的字体大小，它们的值该怎么写？

答案：<h1> 应该是1.7em，而 <h2>是 1.3em。

测试字体大小

这是改进了的日志，字体比以前小一些，看看有什么变化……



there are no Dumb Questions

问：在<body>元素中定义字体大小，也就是定义了网页的默认字体大小，是吗？这是为什么？

答：是的。在<body>元素中定义字体大小，就可以把其他元素的字体大小定义为与它们的父元素相关联的大小。这有什么好处呢？如果你想改变字体大小，只需改变body字体大小，别的都会以适当的比例改变。

问：要考虑用户调整浏览器字体大小吗？我好像从来没考虑过。

答：是的。几乎所有的浏览器都支持用户调整网页字体大小，许多用户都觉得这个特性方便。如果你用相关联的方式定义字体大小，用户就可以顺利做到这一点。要注意别用像素指定大小，因为有些浏览器不支持用像素调整字体。

问：我喜欢用像素指定字体大小的方法，因为那样网页会精确地显示指定的字体大小。

答：的确是这样——用像素指定元素的字体大小后，就给出了每个元素精确的字体大小。但是这么做就使一些使用某种版本的IE用户不能根据自己的需要灵活调整字体大小了。同时对你自己也造成了一些困难，降低了页面的通用性，因为一旦你想把整个页面的字体都增大，就要做许多变动。

问：em和%之间有什么区别？它们看上去好像都一样。

答：它们是做成同一件事的两种不同方法。都提供一种指定与父元素字体大小相关联的字体尺寸的方法。许多人觉得百分数比em容易理解，在CSS中也更具可读性。不过你可以随意选择自己喜欢的方法。

问：如果没有指定字体大小，就只能是默认字体大小吗？

答：是的，默认字体大小因浏览器类别甚至浏览器版本不同而不同，但在大多数情况下默认body字体大小是16像素。

问：默认标题字体大小是多大？

答：这也是取决于浏览器，但一般说来，<h1>是默认body字体大小的200%，<h2>是150%，<h3>是120%，<h4>是100%，<h5>是90%，<h6>是60%。默认的<h4>和body字体大小一样，<h5>和<h6>要更小一些。

问：如果指定body字体大小时不用关键字而用em或%，能行吗？假如指定body的font-size为90%，具体指什么呢？是什么的90%？

答：完全可以。如果指定body部分字体大小为90%，就是默认字体大小的90%，即我们刚说过的通常的默认字体大小是16像素，它的90%就是14像素左右。如果不喜欢用关键字指定字体大小，就用em或%吧。

问：好像浏览器在很多方面都有所不同：font-famlily、font-size，不同的默认值等。怎么知道我设计的网页在别的浏览器上看起来是什么效果呢？

答：问得好。最简单的答案是如果你按照这一章中的指导做，你的大多数设计在别的浏览器中看起来会很不错。然而，你应该明白，在不同的浏览器中还是会稍微有所不同——字体会稍微大一些或小一些，每处的空间也会稍有不同等。但是所有的差别都很细微，不会影响网页的可读性。

不过如果你实在想让网页在大多浏览器中看起来都毫无二致，那么就需要在许多浏览器中测试了，要做到精确，还可以找一种CSS“hacks”使不同的浏览器显示一致。如果你想深入了解，没问题，但要花费很多时间关注这方面的内容并且可能会获益甚微。

改变字体粗细

font-weight属性用来控制字体的粗细。如你所知，bold文本看起来要比normal文本颜色深并且稍微粗一些。可以把**font-weight**属性设置为**bold**来得到字体的粗体版本，像这样：

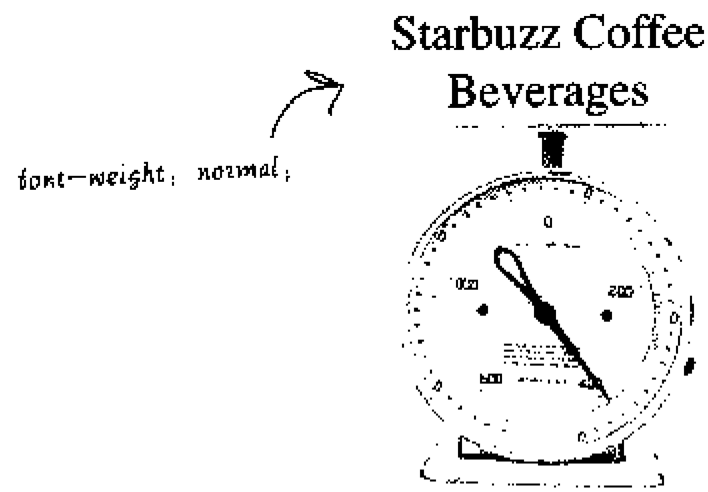
```
font-weight: bold;
```

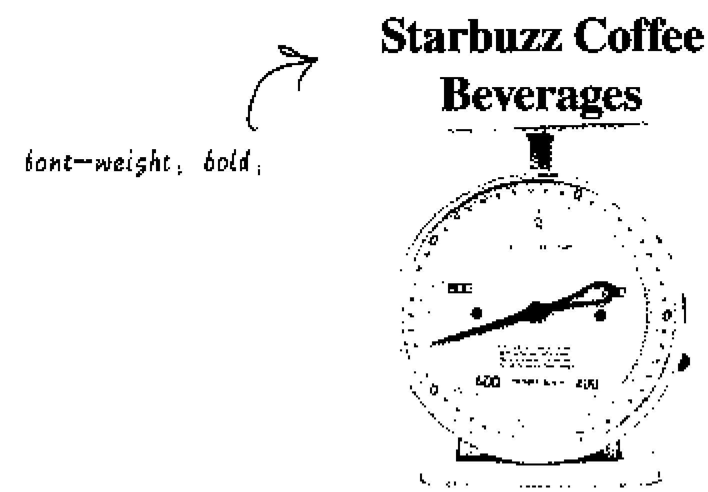
也可以把粗体变回来，如果有一个元素被默认设置为粗体或从父元素继承了粗体，就可以这样把粗体去掉：

```
font-weight: normal;
```

还有两种相对应的**font-weight**属性值：**bolder**和**lighter**。可以用它们使文本的字体较父元素字体更粗或更细。这些值很少用，因为不会有太多字需要考虑粗细方面的细微差别，实际上这两个值也没有太有明显的效果。

也可以把**font-weight**属性值设置为100～900之间的一个数，但还是会有浏览器或字体不支持这种方法，所以这种方法并不常用。





Sharpen your pencil

写CSS，把Tony网页中的标题默认值bold改为normal，把规则加进CSS后，测试一下。答案在下一页。

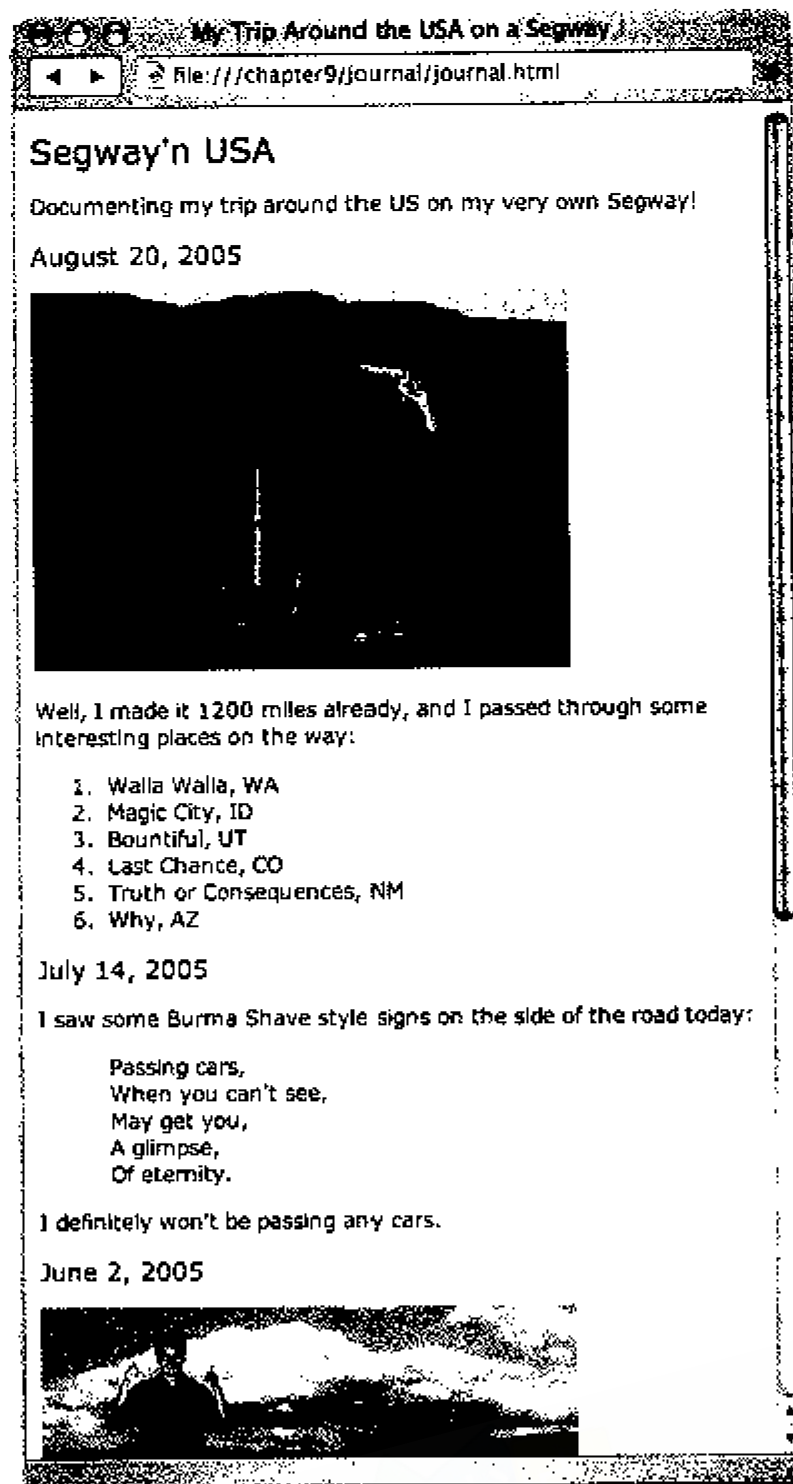
测试normal值的标题

把标题<h1>和<h2>的**font-weight**值改为normal的CSS是这样的：

```
body {
        font-family: Verdana, Geneva, Arial, sans-serif;
        font-size: small;
}
h1, h2 {
        font-weight: normal;
}
h1 {
        font-size: 170%;
}
h2 {
        font-size: 130%;
}
```

在同一个CSS规则中把两个标题的font-weight值都改为normal，这种把相同的属性合并在一个CSS规则中以避免重复的用法是很不错的。

这是显示的效果，现在标题看起来淡一些了。



给字体添加样式

你很熟悉italic(斜体）文本，对吗？italic文本是倾斜的，有时多一些弯曲的衬线。例如，比较以下两种格式：

not italic

italic

italic文本向右倾斜而且衬线有弯曲。

可以用font-style属性在CSS中给文本添加斜体（italic）样式：

```
font-style: italic;
```

把"italic"写成"italics"是个很容易犯的错误。如果写错了，就看不到斜体文本，所以别忘记检查拼写。

然而，有的字体并不支持italic（斜体）格式，可以用(oblique)倾斜文本代替。oblique文本也是倾斜的，不过字体不是一些专门设计的倾斜字母，浏览器只是把普通字母倾斜显示了，对比一下这些没倾斜和倾斜了的样式：

not oblique

oblique

在oblique样式中普通字母向右倾斜。

也可以用**font-style**属性得到oblique文本，像这样：

```
font-style: oblique;
```

实际上，你将会发现，由于你选择的字体和浏览器不同，这两种样式有时看起来一样，有时不一样。所以，除非必须区分出italic和oblique，其他情况任选一种就可以。但如果这重要，就需要同时测试字体和浏览器以达到最佳效果。

Italic和oblique是让字体倾斜显示的两种格式。

除非能控制用户使用的字体和浏览器，否则你会发现有时显示的是italic，有时是oblique，指定的是其中的哪种都无所谓。

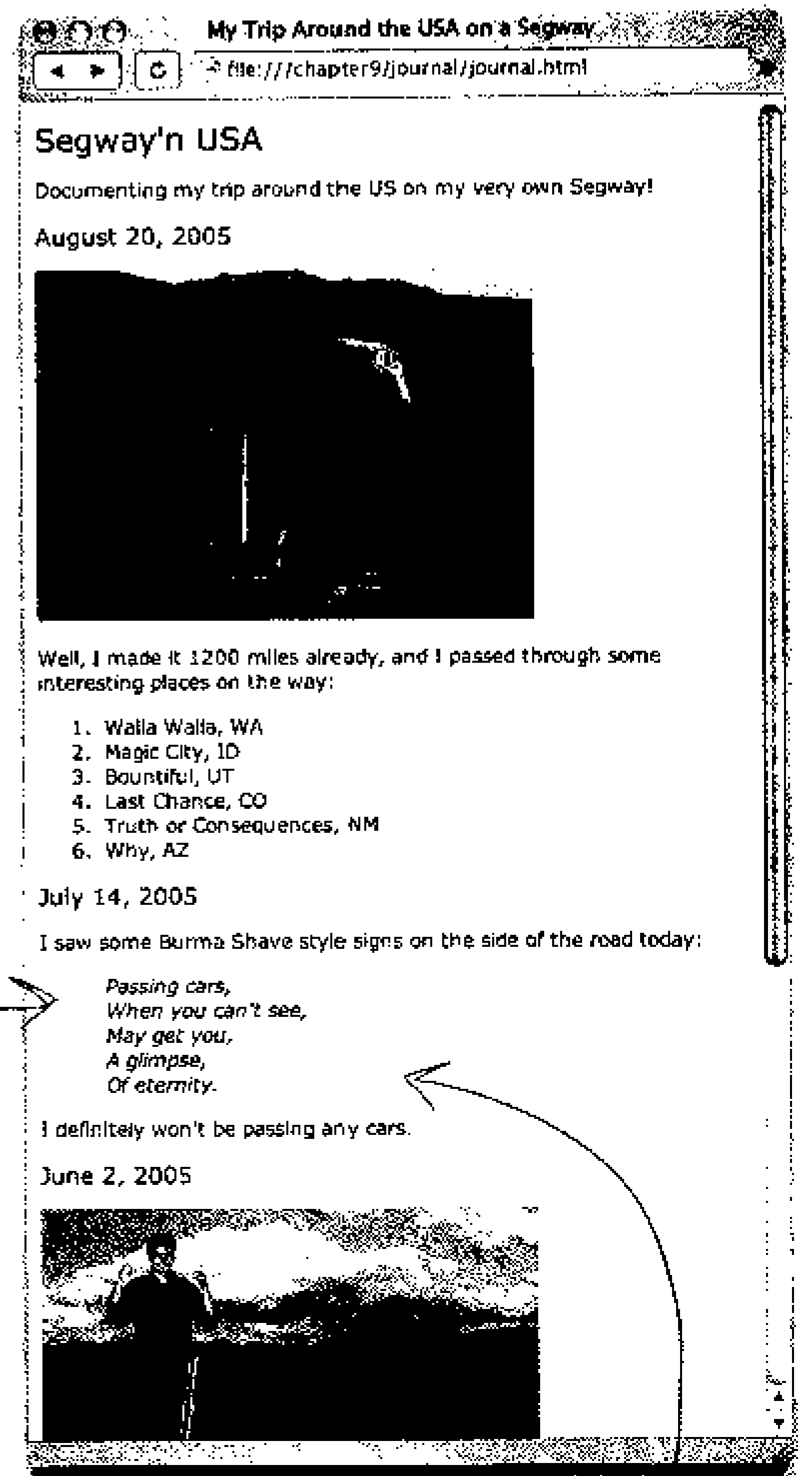
就用italic吧，别担心会有什么不同（你对此也无能为力）。

把Tony网页中的引用变成斜体

我们将用**font-style**属性使Tony的引用变得生动一些。还记得**<blockquote>**中的柏玛刮胡膏口号吧？我们将把标语变成斜体格式以使它在文本中更加醒目。为此我们只需把**<blockquote>**的**font-style**值设为italic，如下：

```
blockquote {
        font-style: italic;
}
```

把这条新的CSS规则加进"journal.css"文件，保存并测试。你会看到标语变成了斜体，这就是测试结果。



这是Tony网页中的柏玛刮胡膏口号的新样式，文本正是我们想要的斜体。

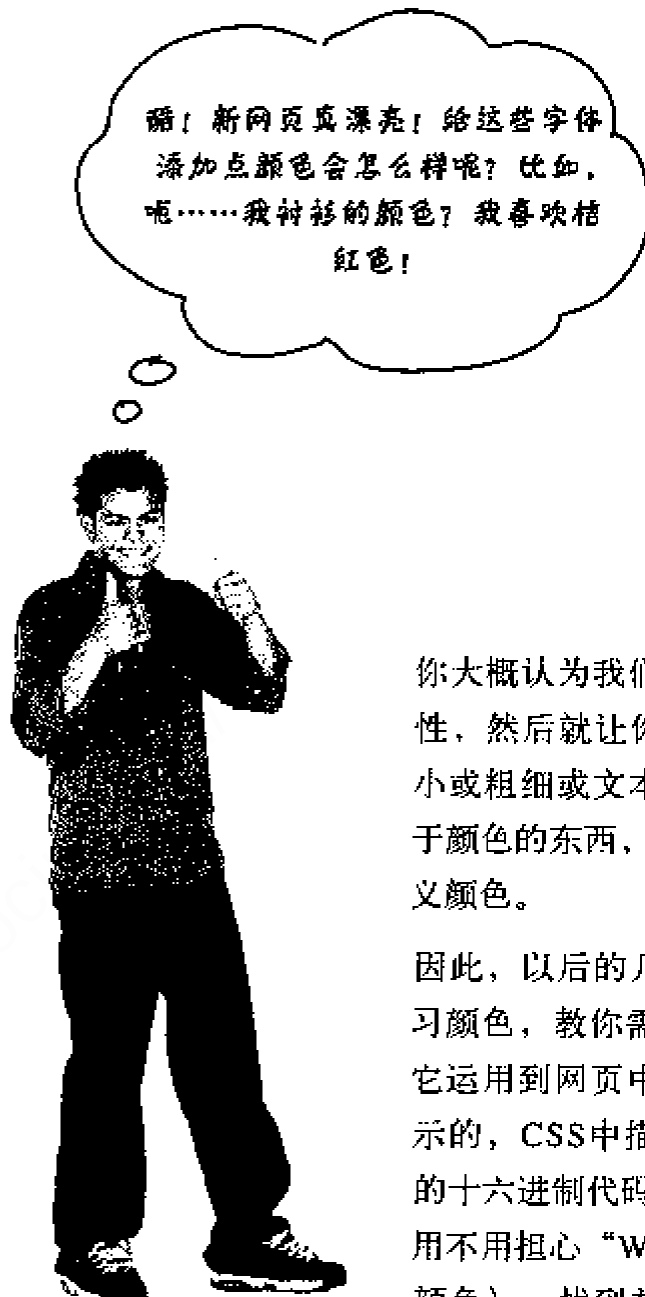
there are no Dumb Questions

问： <blockquote>里还有个<p>，<blockquote>里的文本实际上是<p>里的。那它怎么把段落变成斜体的呢？

答： 别忘了，默认情况下，大多数元素从父元素继承字体格式，本段的父元素是<blockquote>，所以<blockquote>里的段落就继承了斜体格式。

问： 为什么不把文本放到<blockquote>的<em>元素中呢？它不是同样能把<blockquote>变成斜体吗？

答： 别忘了<em>是用来指定结构的，它意味着强调一段文字。给<blockquote>添加样式并不意味着要强调其中的文本。尽管如你所说的，在大多数浏览器中<em>的格式是斜体，但用它给<blockquote>里的文本添加样式是不对的。还要谨记<em>的格式会改变，不一定总是斜体。

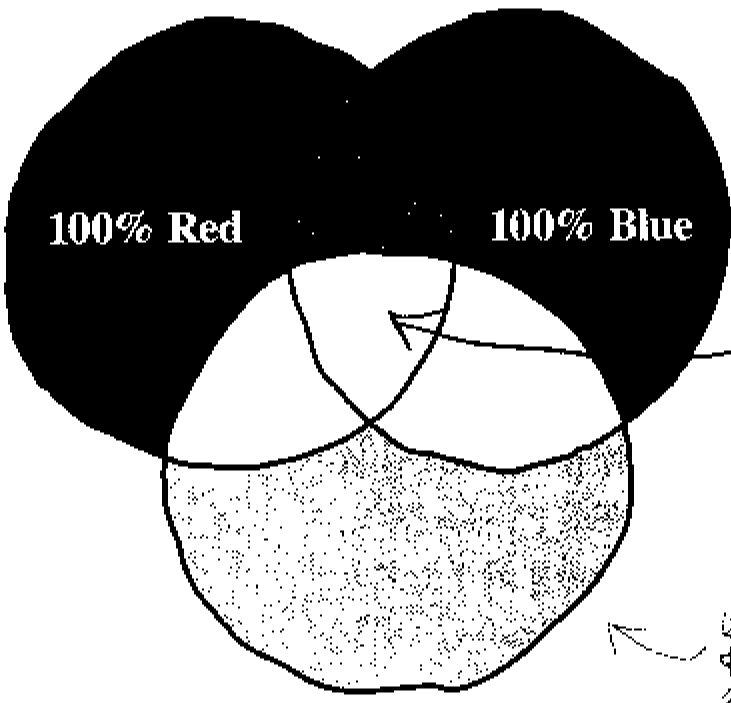


你大概认为我们会告诉你有一个`color`属性，然后就让你用它。但是，不像字体大小或粗细或文本样式，你需要懂得很多关于颜色的东西，才能很好地用它在CSS中定义颜色。

因此，以后的几页中，我们将带你深入学习颜色，教你需要知道的所有东西得以把它运用到网页中：颜色是如何在屏幕上显示的，CSS中描述颜色的几种方法，神秘的十六进制代码（hex codes）是怎么回事，用不用担心“Web safe colors”（网页安全颜色），找到并指定颜色最简单的方法是什么等。

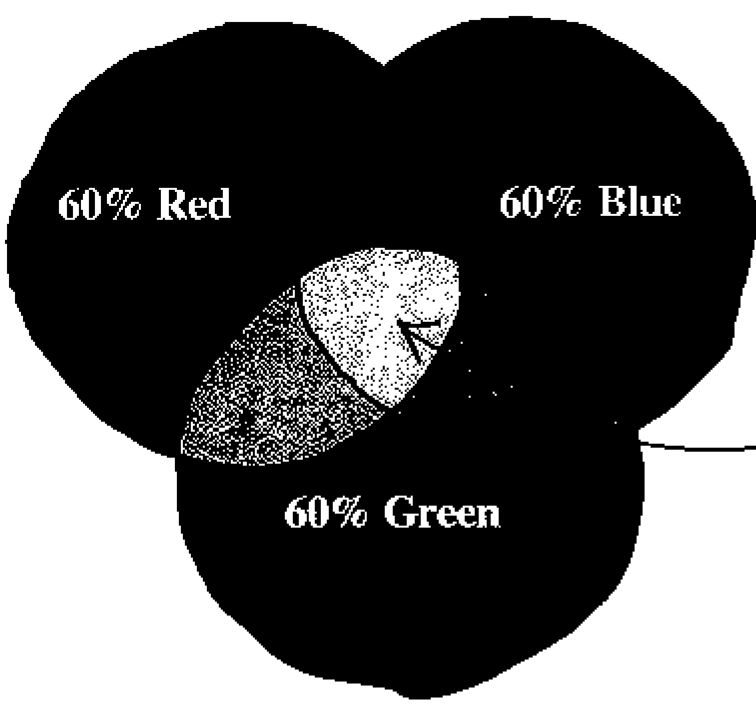
Web颜色如何工作？

你会发现网页上有许多地方可以添加颜色：背景颜色，边框颜色，还有字体颜色等。但是，电脑上的颜色是如何工作的呢？我们来看一下。



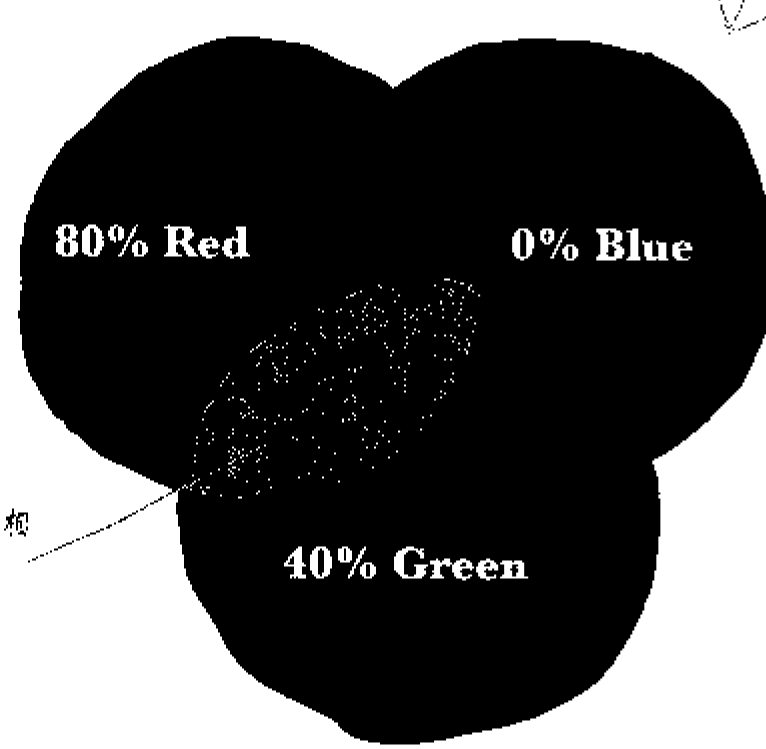
网页颜色是根据红色、绿色和蓝色三原色以一定比例组成来指定的。可以把每种颜色指定为从0~100%的一个数，然后混合起来就组成了一种颜色。例如，如果把红色100%、绿色100%和蓝色100%混合起来，就得到白色。注意在电脑屏幕上，混合得到的颜色会比较亮，毕竟是我们混和出来的光。

这是把红、绿和蓝相混合，看一下中间就知道它们三者相混合是什么样了。



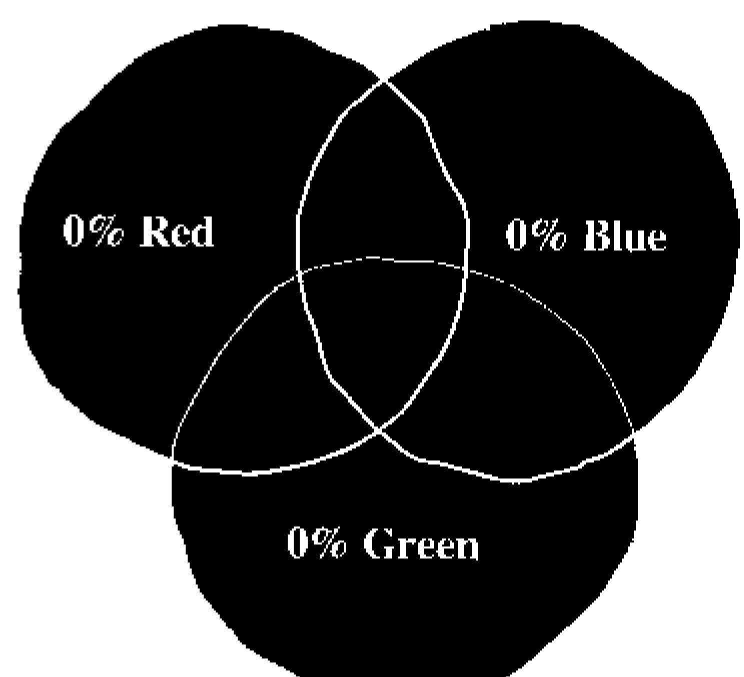
但是如果每种成分只加60%会得到什么结果呢？不够白是吗？换言之，得到的是灰色，因为我们只是给屏幕加了相同数量的三种颜色，而没有足够的光。

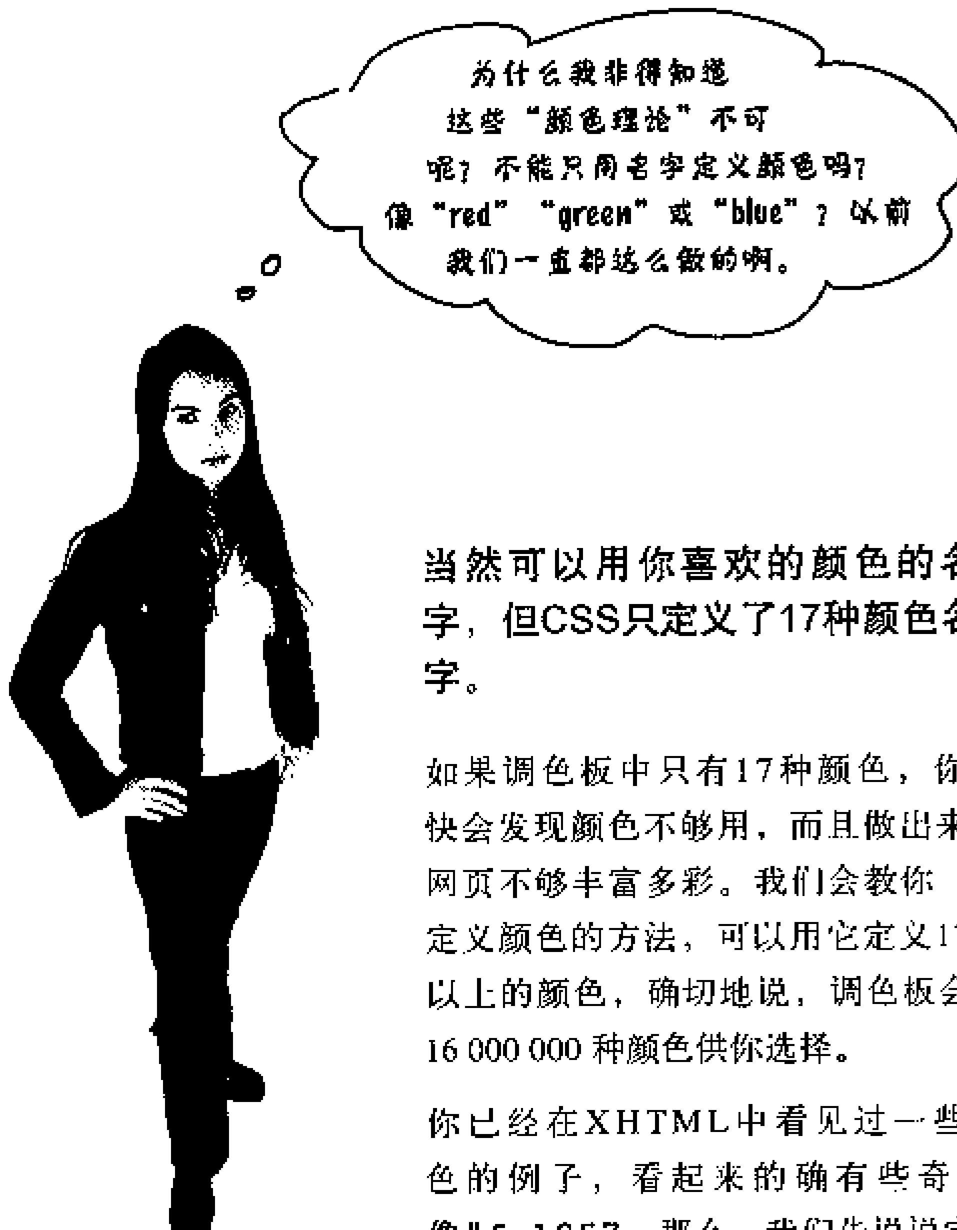
把80%的红色和40%的绿色混合起来，就能得到你想要的桔红色。注意如果某一种颜色没加进去，它就不会影响另外两种颜色。桔红色是因为没有把蓝光加进红色和绿色中而形成的。



www.docin.com

如果混合红色0%、绿色0%和蓝色0%，会得到什么结果？那意味着没给屏幕加任何光，所以得到的是黑色。





当然可以用你喜欢的颜色的名字，但CSS只定义了17种颜色名字。

如果调色板中只有17种颜色，你很快会发现颜色不够用，而且做出来的网页不够丰富多彩。我们会教你一种定义颜色的方法，可以用它定义17种以上的颜色，确切地说，调色板会有16 000 000 种颜色供你选择。

你已经在XHTML中看见过一些颜色的例子，看起来的确有些奇怪，像`#fc1257`。那么，我们先说说定义颜色的方法，然后你就明白如何使用颜色表或图片处理程序轻松地选择颜色了。

如何指定Web颜色？方法有……

CSS提供了好几种指定颜色的方法。可以用颜色名直接定义，也可以根据红、绿、蓝的相对百分比来定义颜色，还可以用十六进制代码定义。十六进制代码是描述红、绿、蓝成分多少的简略表达法。

也许你会觉得Web现在应该有一种统一的格式，然而所有这些格式都很常用，所以全部都知道比较好一些。到目前为止，十六进制代码是指定Web颜色最常用的一种方法。不过要记住所有这些指定颜色的方法的本质都是告诉浏览器加入颜色中的红色、绿色、蓝色成分是多少。

我们将逐一学习CSS中定义颜色的各种方法。

用名字定义颜色

在CSS中描述颜色最直接的方法是用颜色名。但是，如你所知，用这种方法只能定义17种颜色。假如想把body元素的背景颜色定义为“silver”（银白色），你在CSS中可以这么写：

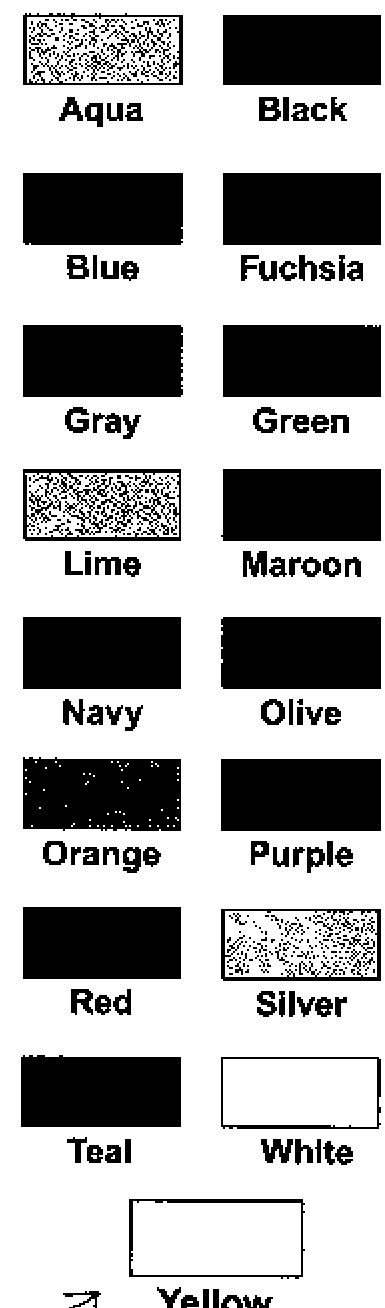
```
body {
        background-color: silver;
}
```

这是body规则。

background-color属性。

写颜色名。

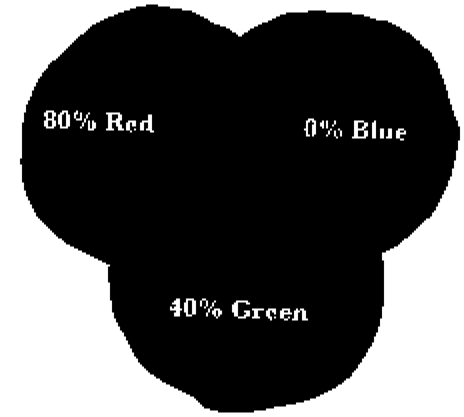
所以，用名字定义颜色，只需用输入颜色名字作为属性值。CSS颜色名字不区分大小写，所以输入silver、Silver或SILVER都是一样的。这些是CSS中预先定义的17种颜色。记住这些名字只是预先定义了红色、绿色和蓝色三原色的比例。



书上的颜色是印刷的页面反射光线形成的；在电脑上，光是屏幕发出的，所以这些颜色在网页中看起来会稍微有些不同。

用红色、绿色和蓝色值定义颜色

同样，也可以用红色、绿色和蓝色的各自比例定义颜色。假如你想定义桔红色，看一下前两页，是由红色80%、绿色40%和蓝色0%组成的。你可以这么做：




```
body {
      background-color: rgb(80%, 40%, 0%);
}
```

以"rgb"（red green blue的缩写）开始。

然后在圆括号内定义红色、绿色、蓝色的百分数，每个数字后跟一个%。

也可以用0~255之间的某个数值定义红色、绿色和蓝色的值。所以，可以用红色204、绿色102和蓝色0代替红色80%、绿色40%和蓝色0%。

这些数字是怎么来的？255的80%是204，255的40%是102，255的0%是0。

可以这样直接用数值定义颜色：

```
body {
      background-color: rgb(204, 102, 0);
}
```

还是以"rgb"开头。

定义数值就只打数字，不用加%。

there are no Dumb Questions

问：为什么会有两种指定rgb值的不同方法呢？百分数看起来不是更直接一些吗？

答： 有时百分数是更直接一些，但用0～255之间的数字也是有原因的。这个数字跟一个字节的信息包含的值有关。所以，由于历史和技术原因，255经常被用作在一个颜色中定义红色、绿色和蓝色值的计量单位。事实上，你也许已经注意到图片处理程序通常允许用0~255的值定义颜色（如果没有，稍后你会知道怎么做）。

问：我从来没见过有人在CSS中用rgb或实际的颜色名字。好像大家都用#00fc9a这种颜色代码。

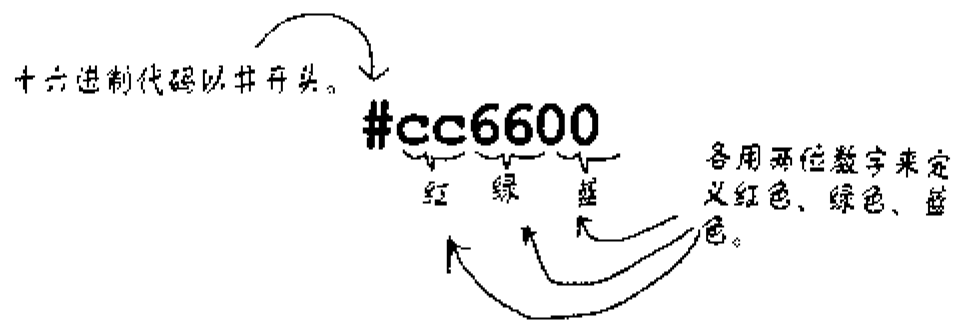
答： 用rgb百分数或数值正变得越来越普遍，但你说对了，"hex codes"仍然得到最广泛的使用，因为人们觉得用这种方法定义颜色很方便。

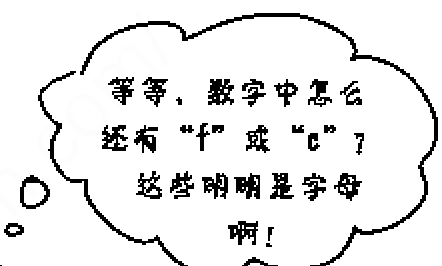
问：有必要一看到rgb（100，50，200）这样的表示就知道它代表什么颜色吗？

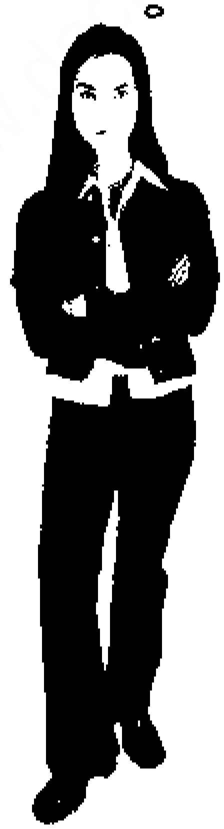
答： 不必。知道rgb（100，50，200）代表什么颜色的最好方法是载入浏览器或用图片处理程序来查看。

用十六进制代码定义颜色

现在我们来解决那些看起来很神秘的十六进制代码。关于它们的秘密是这样的：每两位数字代表红色、绿色、蓝色中的一种颜色。前两位数字代表红色，中间两位代表绿色，最后两位代表蓝色。像这样：







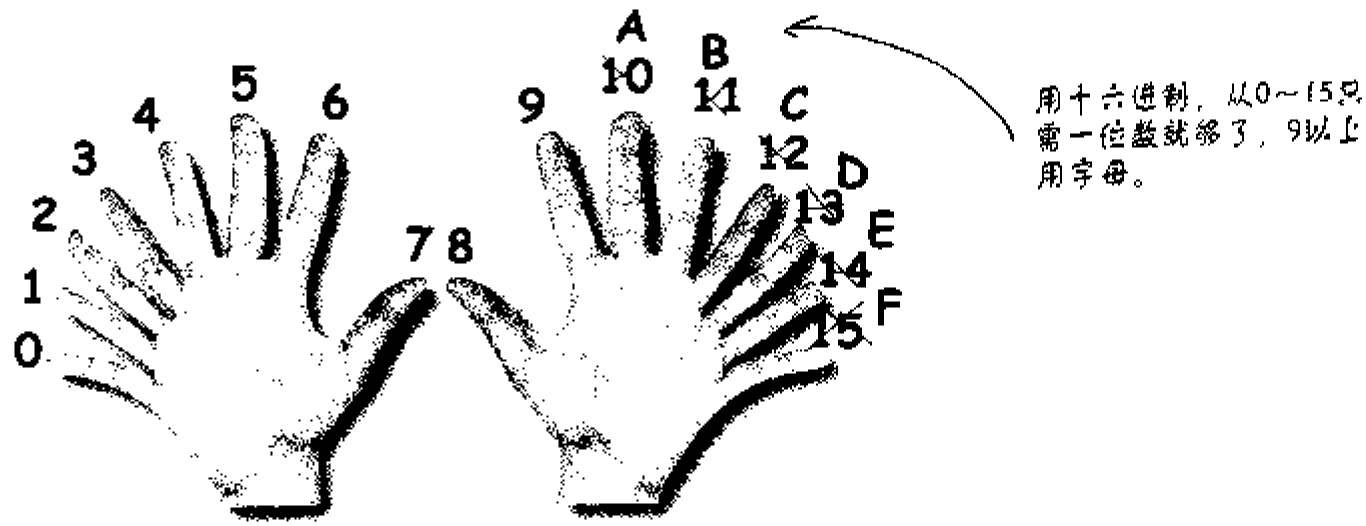
不管你信不信，它们的确是数字，不过是用只有计算机科学家才会喜欢用的符号写成的。

好了，告诉你十六进制代码的另一个秘密：每一组两位数代表一个0～255的数字（好像听过？）。问题是如果用数字，用两位只能代表0～99，不是吗？科学家们不想被简单的几个0～9这样的数字限制住，于是想到借助于一些字母（A到F）就可以代表255个值了。

我们快速看一下十六进制代码是如何工作的，然后就教你如何从颜色表或图片处理程序中得到它们。

快速了解十六进制代码

关于十六进制代码你首先需要知道的是，它们不是基于10个数（0~9），而是基于16个数（0~F）的。十六进制数是这样的：

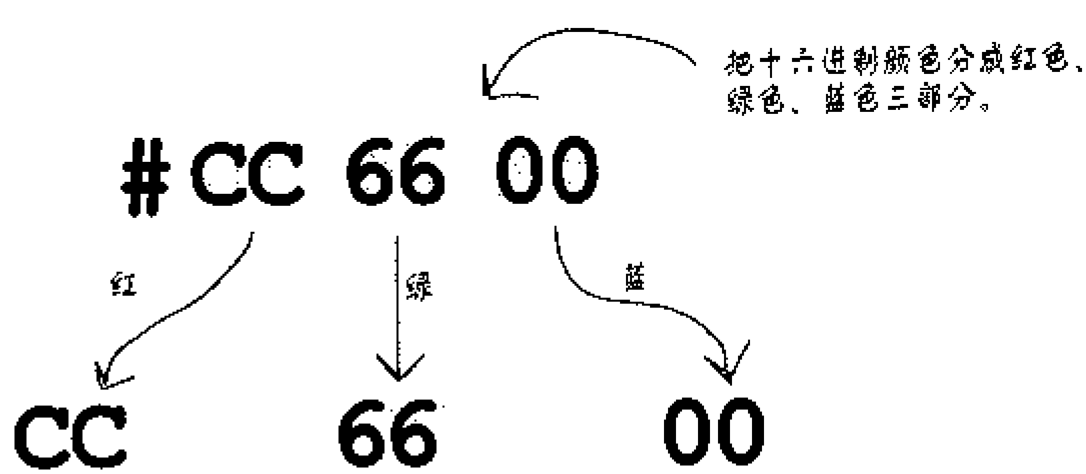


所以如果看到十六进制数B，就知道它代表11。但是BB，E1或FF代表什么数呢？我们把一个十六进制颜色拆开，看它真正代表什么。以后你遇到十六进制颜色都可以这么做。

第一步：

把十六进制颜色分成三部分。

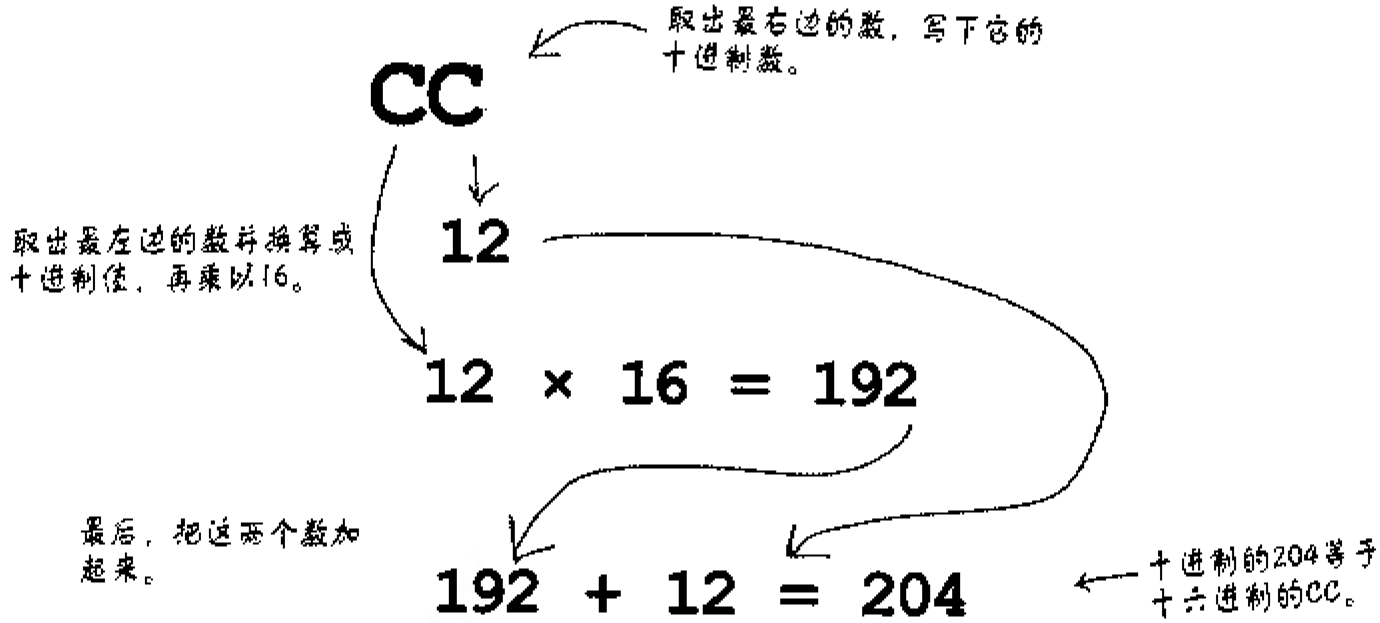
记住每个十六进制颜色都由红色、绿色、蓝色三部分组成。首先要把它们分开。



第二步：

把每个十六进制数转化成相应的十进制数。

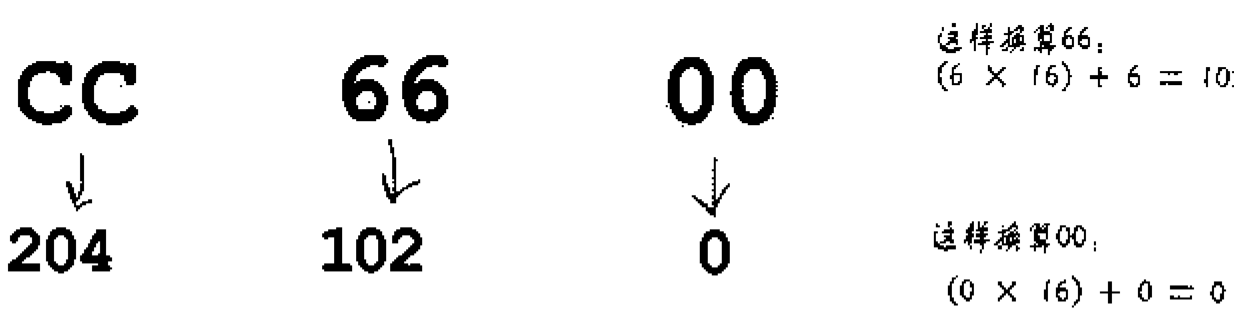
现在已分成三部分，每部分的值都可以换算成0~255的一个数。我们从红色的十六进制数开始。



第三步：

别的两个值也这么换算。

重复同样的方法换算另两个值，结果是这样的：

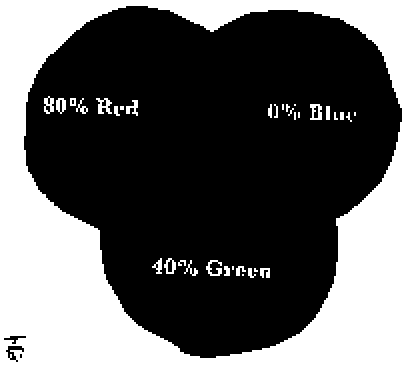


第四步：

没有第四步，完成了！

就这些了。得到每部分的数字就知道一种颜色中到底加入了多少红色、绿色和蓝色了。可以用同样的方法把任何十六进制代码拆开，现在来看一下如何确定网页颜色。

总结一下：



现在你已经学了几种定义颜色的不同方法。以红色80%、绿色40%和蓝色0%组成的桔红色为例，CSS中可以用以下这些方法中的任意一种来指定这种颜色：

```
body {
        background-color: rgb(80%, 40%, 0%);
}
```

用红色、绿色和蓝色的百分数定义。

```
body {
        background-color: rgb(204, 102, 0);
}
```

用0～255的数字定义红色、绿色和蓝色的数量。

```
body {
        background-color: #cc6600;
}
```

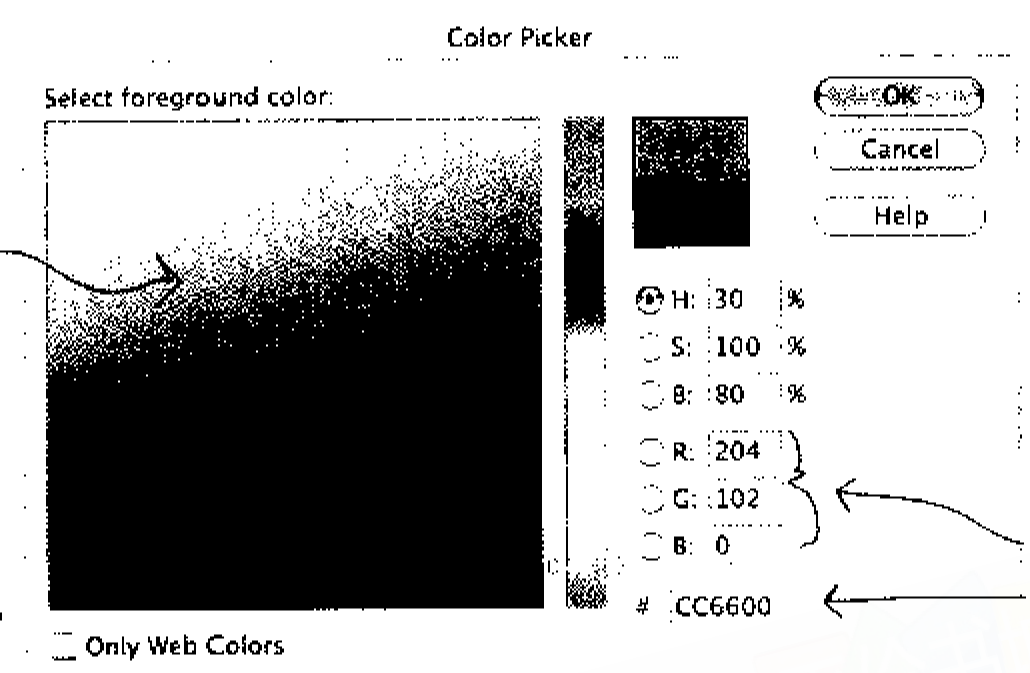
用简单的十六进制代码定义。

如何确定Web颜色

确定网页颜色最常用的两种方法是用颜色表或像Photoshop Element这样的程序。你还会发现许多网页允许你选择颜色并且在rgb和十六进制代码之间转换。我们看一下Photoshop Element（大多数图片编辑器提供同样的功能）。

大多数图片编辑器都有一个调色板，允许你通过用一种或多种颜色的光谱来选择颜色。

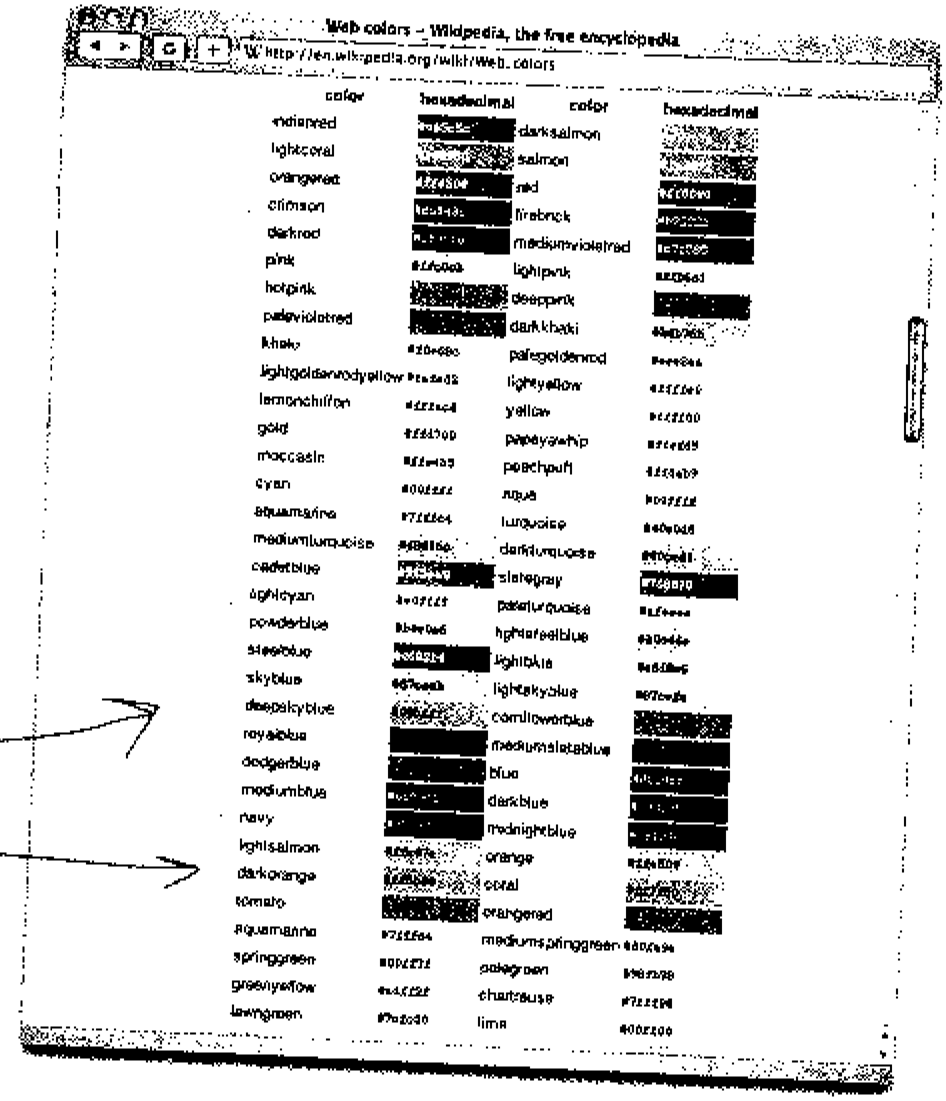
调色板也允许你只选择"Web safe"颜色，很快我们会讲到这个。



一旦你选择了一种颜色，调色板就会同时以rgb值和十六进制代码显示出来。

使用在线颜色表

你还能在网上找到一些有用的颜色表。这些表根据许多不同的标准来显示具有相应的十六进制代码的网页颜色。要使用表中的颜色，你只需要选择网页中用到的颜色，然后把它的十六进制代码复制到CSS中就可以了。



这个表在http://en.wikipedia.org/wiki/Web_colors上。还可以通过搜索“HTML color charts”找到更多颜色表。

颜色一定要用十六进制代码，不要用名字，因为有些浏览器不支持名字。

there are no Dumb Questions

问： **听说如果不用网页安全颜色，网页在别的浏览器中就不能正确显示。我们为什么没讲到网页安全颜色呢？**

答： 早期的网页浏览器，支持大量颜色的电脑屏幕很少，因此创建了网页安全调色板来保证网页在大多数显示器中显示一致。

如今图片发生了巨大的变化，而且大多数Web用户的显示器能支持上百万种颜色。所以，除非你知道有一些特殊的用户显示颜色受限制，否则就可以把“网页安全颜色”当作历史来看待了。

问： **我现在知道怎么指定颜色了，但是不知道如何搭配字体颜色。**

答： 回答好这个问题要用一整本书，但有一些选择字体颜色的基本指南。最重要的是文本和背景用高对比的颜色以提高可读性。例如，白背景黑文本就是最高的对比。也不必总是用黑和白，但最好做到文本用深颜色，背景用浅颜色。有些颜色一起用会产生很奇怪的视觉效果（像蓝色和桔红色，红色和绿色），所以在发布前最好让朋友先看看颜色搭配的效果。

问： **我还见过像#cb0这样的十六进制代码，它代表什么呢？**

答： 如果每两位数字都一样，就可以简写。例如，#ccbb00可以缩写成#cb0，或#11eeaa写成#1ea。但如果十六进制代码是#ccbb10这样的，就不能缩写。



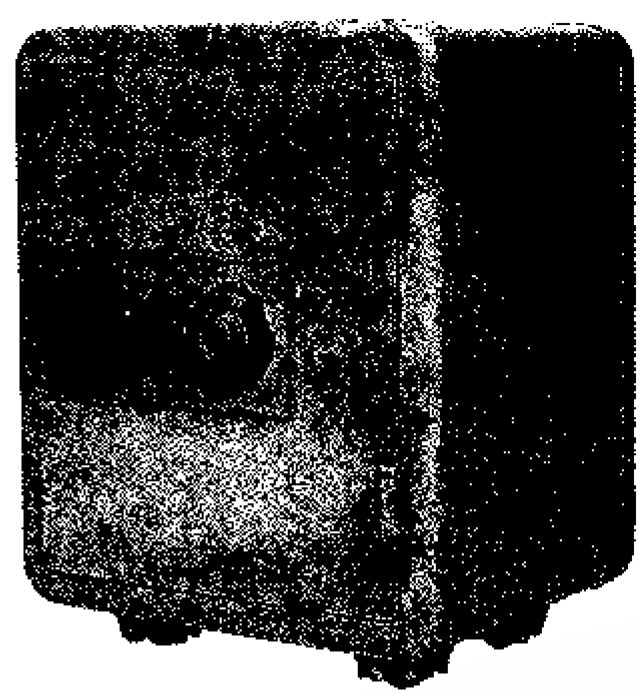
挑战 打开保险柜

Evel博士的主设计图锁在他的私人保险柜中，你只得到一个提示：他把密码编成十六进制代码，为了不忘记密码，把这个代码做成了自己主页的背景颜色。你的工作是破解十六进制代码，找出保险柜的密码。为此，只要把他的网页颜色转换成红色、绿色和蓝色的十进制值，就能得到密码，顺序是右－左－右。以下是他主页的背景颜色：

```
body {
      background-color: #b817e0;
}
```

破解代码，把密码写在下面：

右 ________　　左 ________　　右 ________



回到Tony的页面……我们将把标题变为桔红色，并添加下划线

现在有关颜色的所有内容都讲完了，现在就来给Tony的网页添加点颜色吧。他想要桔红色，那就给他来点桔红色。但是不能把所有的文本都变成桔红色，那样网页就可能太单调，而且在白色背景上也不好读，那么只给标题添加一些装饰性的色彩就可以了。桔红色的深度足够显出背景和文本之间的对比，也能跟照片中的桔红色（Tony衬衫的颜色）相互搭配。我们在标题和照片之间建立一种使图像和文本相互对应的颜色关系。为了确保标题突出且日志项之间有间隔，还要给每个标题加下划线。还没学过怎么添加下划线，但先照着做吧，接着就会讲到文本修饰。

以下是要在CSS中修改的，在“journal.css”中做这些改动。

```
body {
        font-family: Verdana, Geneva, Arial, sans-serif;
        font-size: small;
}

h1, h2 {
        font-weight: normal;
        color: #cc6600;
        text-decoration: underline;
}

h1 {
        font-size: 170%;
}

h2 {
        font-size: 130%;
}

blockquote {
        font-style: italic;
}
```

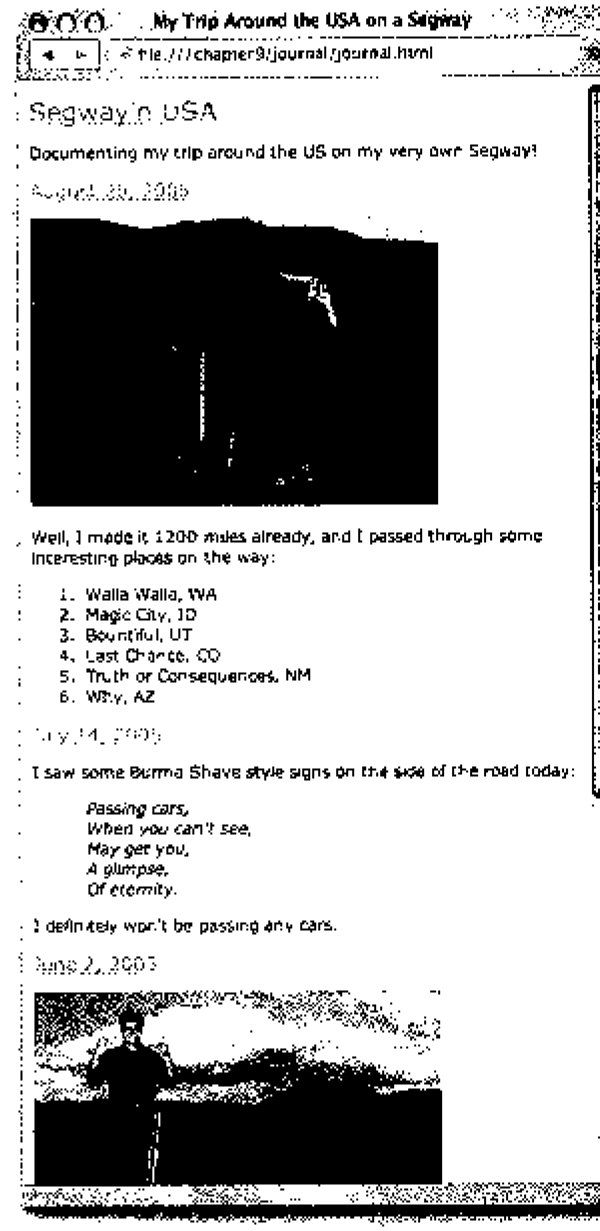
我们将把<h1>和<h2>都变成桔红色，所以把颜色属性放在同一个规则中。

这是Tony想要的桔红色的十六进制代码，换成百分数就是rgb(80%,40%,0%)。

这是添加下划线的规则，使用text-decoration属性并设置其值为underline。

测试Tony的桔红色标题

修改文件“journal.css”，把color属性加进“h1，h2”规则之中，完成后重新加载网页并仔细观察结果。



<h1>和<h2>都变成了桔红色，与照片中的背景和衬衫颜色搭配得很好。

使用下划线使标题显得更突出了，我们觉得这是区分标题的好方法，但实际上它们看起来太像可点击的链接了，因为人们总认为网页中带下划线的内容是可点击的。

所以，下划线好像不是一个好主意。我们快看看其他文本修饰，然后重新考虑网页中的这些下划线。

Sharpen your pencil

这些颜色都有什么共同点？用它们做字体颜色在网页中试一下，也可以用图片处理程序的调色板看一下它们是什么颜色（把十六进制代码直接输入对话框即可）。

#111111	#444444	#777777	#aaaaaa	#dddddd
#222222	#555555	#888888	#bbbbbb	#eeeeee
#333333	#666666	#999999	#cccccc	

用很少的篇幅告诉你想知道的关于text-decorations的所有内容

文本修饰允许你在浏览器、闪烁文本中添加一些underlines、overlines、line-throughs(也叫作strike-through)之类的修饰效果。要添加文本修饰，只要在元素中设置text-decorations属性便可，就像这样：

```
em {
      text-decoration: line-through;
}
```

这条规则将会使元素<em>的文本中央有条线穿过。

可以同时设置几个decoration值。假如你想同时有underline和overline的修饰效果，可以像这样定义文本修饰：

```
em {
      text-decoration: underline overline;
}
```

这条规则使元素<em>既有underline，又有overline。

如果文本继承了你不想要的文本修饰，用值“none”就可以去掉。

```
em {
      text-decoration: none;
}
```

采用这条规则，文本就没有修饰了。

---

there are no Dumb Questions

问：如果元素<em>有两条不同的规则，一条定义overline，一条定义underline，能把它们放在一起来得到两种修饰吗？

答：不能。需要把两个值合并在一条规则中才能得到两种文本修饰。text-decoration只能选择一条规则，不能同时选择两条独立的规则。text-decoration样式所选择的规则决定了所用的修饰，所以得到两种修饰的唯一方法是在同一个text-decoration声明中指定它们。

问：为什么颜色属性不叫text-color?

答：颜色属性实际上控制的是元素的前景颜色，所以它能控制文本和边框的颜色，不过可以用border-corlor属性给边框定义它自己的颜色。

问：我喜欢line-through。能用它来指定编辑的文本中要删除的内容吗？

答：可以，不过还有更好的方法。XHTML有一个我们没讲过的元素叫做<del>，它可以标明XHTML中应该被删掉的内容。同样，有一个叫做<ins>的元素，可以标明应该插入的内容。一般浏览器会分别用strike-through和underline样式化这些元素。用CSS可以选择任何你喜欢的方式。用<del>和<ins>，不只是样式化元素，而且标明了内容的意义。

问：什么时候可以看到闪烁(blink)修饰？

答：闪烁是旧版Netscape样式的延续。浏览器不需要应用它，大多数人觉得用闪烁是很糟糕的。所以根本就不必理会它。

去掉下划线……

我们去掉容易让人误解的下划线，代之以下边框，像在休闲室中做的那样。为此，打开文件“journal.css”，在“h1,h2”合并规则中做以下改动：

```
h1, h2 {
        font-weight: normal;
        color: #cc6600;
        border-bottom: thin dotted #888888;
        text-decoration: underline;
}
```

在元素<h1>和<h2>下添加边框。几乎可以当作英语来读它：“add a thin, dotted line with the color #888888 on the bottom border”……

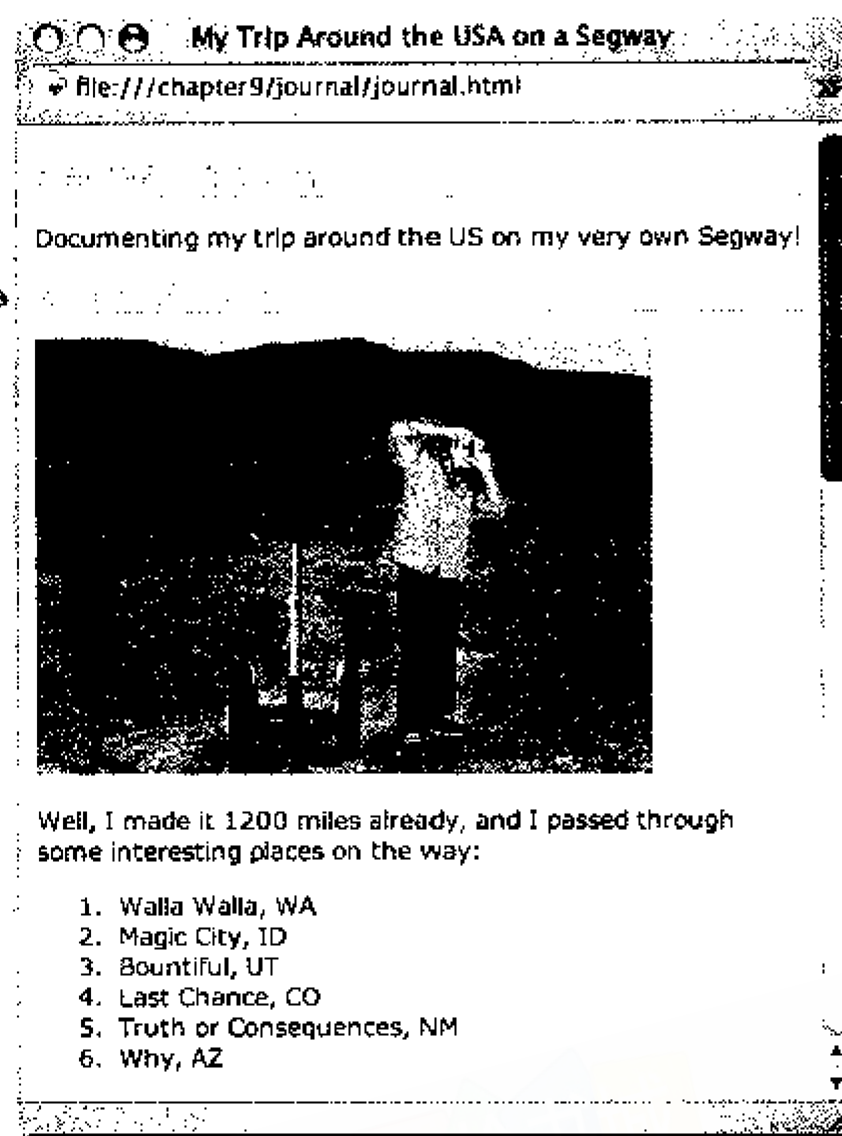
……下章我们将深入学习边框。坚持，就快到了！

删除原来的文本修饰。

新的“下划线”看起来是这样的，绝对要比文本修饰下划线美观，而且不会迷惑人。

元素<h1>和<h2>下有了边框，而不是下划线。

注意边框一直延伸到页面边缘，而不只是在文本底下。为什么呢？下一章中会告诉你答案。



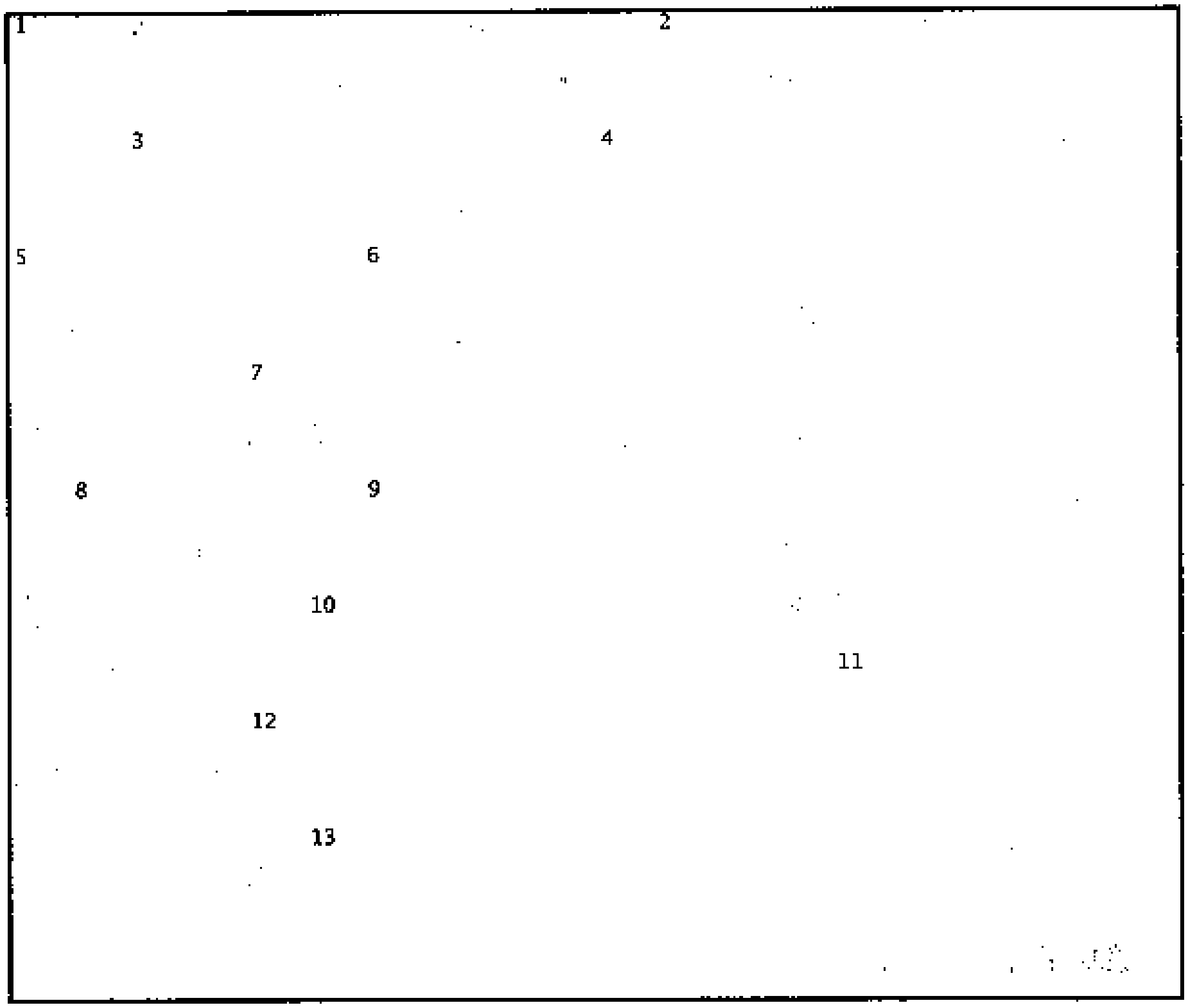
要点

- CSS提供许多属性来控制字体外观，包括font-family,font-weight,font-size和font-syle。
- 一个font-family是一系列有共同特征的字体。
- 网页的字体系列有serif，sans-serif，monospace，cursive和fantasy。serif和sans-serif最常用。
- 访问者在你的网页上看到的字体格式取决于其电脑中安装的字体。
- 比较好的做法是在CSS属性font-family中定义多个字体以防用户未安装首选字体。
- 最后一种字体通常是一种通用字体，如serif或sans-serif。当浏览器没找到前几种时可以找合适的来代替。
- 用px、em、%或关键字定义font-size。
- 如果用像素（“px”）定义字体大小，就是告诉了浏览器字母的高度为多少像素。
- em和%是相对字体大小，所以用em和%定义字体大小意味着字母的字体大小将是父元素字体大小的相对值。
- 用相对大小定义字体大小可以使网页具有更强的通用性。
- 在body规则中用字体大小关键字来设置基字体大小，那么如果用户想把文本变大或变小，所有的浏览器都可以按比例来缩放字体大小了。
- 用font-weight属性可以将文本加粗。
- font-syle属性用来创建italic或oblique文本，italic和oblique修饰的文本是倾斜的。
- 网页颜色是通过把不同数量的红色、绿色和蓝色相混合产生的。
- 如果把红色100%、绿色100%和蓝色100%混合，就得到白色。
- 如果把红色0%、绿色0%和蓝色0%混合，就得到黑色。
- CSS有17种预定义的颜色，包括黑色、白色、红色、蓝色和绿色。
- 可以用红色、绿色和蓝色的百分数，也可以用0～255的数字给出红色、绿色和蓝色的值或者用颜色的十六进制代码来定义任何你想要的颜色。
- 要找到你想要的一种颜色的十六进制代码，用图形编辑器的调色板或某种在线网页工具，轻而易举就能得到。
- 十六进制代码有6位数，每一位可以是0～F中的一个。前两位代表红色的数值，中间两位代表绿色的数值，最后两位代表蓝色的数值。
- 可以用text-underline属性给文本添加下划线。添加下划线的文本容易被用户当作链接文本，所以使用这个属性要谨慎。



XHTML填字游戏

你在本章中学到了很多：字体、颜色、粗细和格式。现在做另一个填字游戏，以加深对知识的理解与巩固。



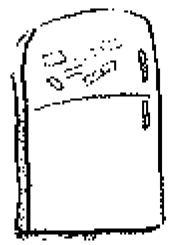
横排提示：

3. 十六进制代码用几个不同的数字。
4. 像#111111~#EEEEEE这样的颜色都是___色调。
5. 相似的字体组成________。
7. 不能很好地处理像素定义的字体大小的浏览器。
8. 可以用来标识要删除的文本的元素。
9. em和%都是这种类型的大小。
10. 网页中几乎从来不用的字体系列。
12. 控制字体粗细。
13. underline和line-through是text________的例子。

竖排提示：

1. 没什么衬线的字体。
2. 可以用像素、em或________定义字体。
3. 在电脑屏幕上看起来清晰一些并且容易读一些。
6. 在font-family属性中定义字体时，你要定义_________。
11. 从来不会用到的一种文本修饰。

字体帖解答



把下面的字按字体归类，将左边方框中的单词按类别填进右边对应的字体系列下，核对完答案后再开始学习下一页内容。以下是答案。

Monospace Family

Messenger

Bainbridge

Fantasy Family

Crush

Cursive Family

CARtOON

Sans-serif Family

Iceland

Angel

Nautica

Serif Family

Savannah

Quarter

Palomino



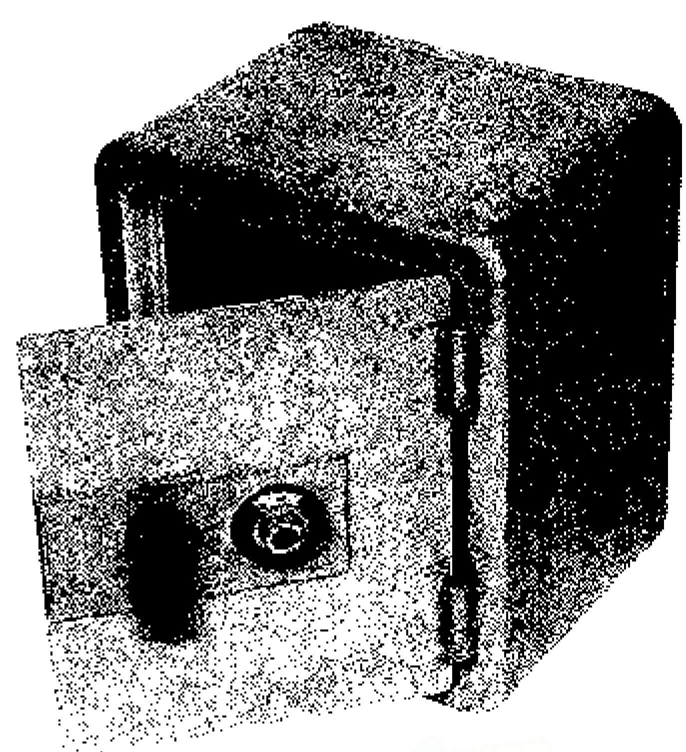
打开保险柜答案

Evel博士的主设计图锁在他的私人保险柜中，你只得到一个提示：他把密码编成十六进制代码，为了不忘记密码，把这个代码做成自己主页的背景颜色。你的工作是破解十六进制代码，找出保险柜的密码。为此，只要把他的网页颜色转换成红色、绿色和蓝色的十进制值，就能得到密码，顺序是右－左－右。这是他的主页的背景颜色：

```
body {
        background-color: #b817e0;
}
```

破解代码，把密码写在下面：

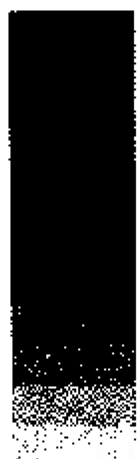
右 (11 * 16) + 8 = 184　　左 (1 * 16) + 7 = 23　　右 (14 * 16) + 0 = 224



Sharpen your pencil 答案

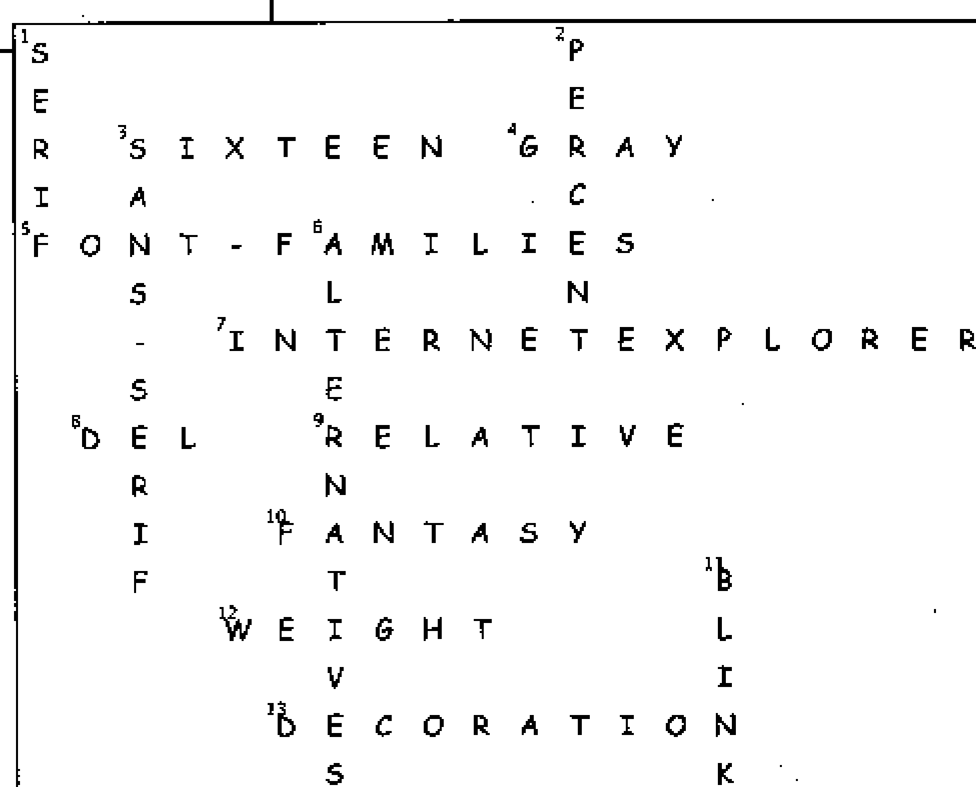
这些颜色有什么共同点？用它们作字体颜色在网页中试一下，也可以用图形编辑器的调色板看一下它们是什么颜色（把十六进制代码直接输入对话框即可）。

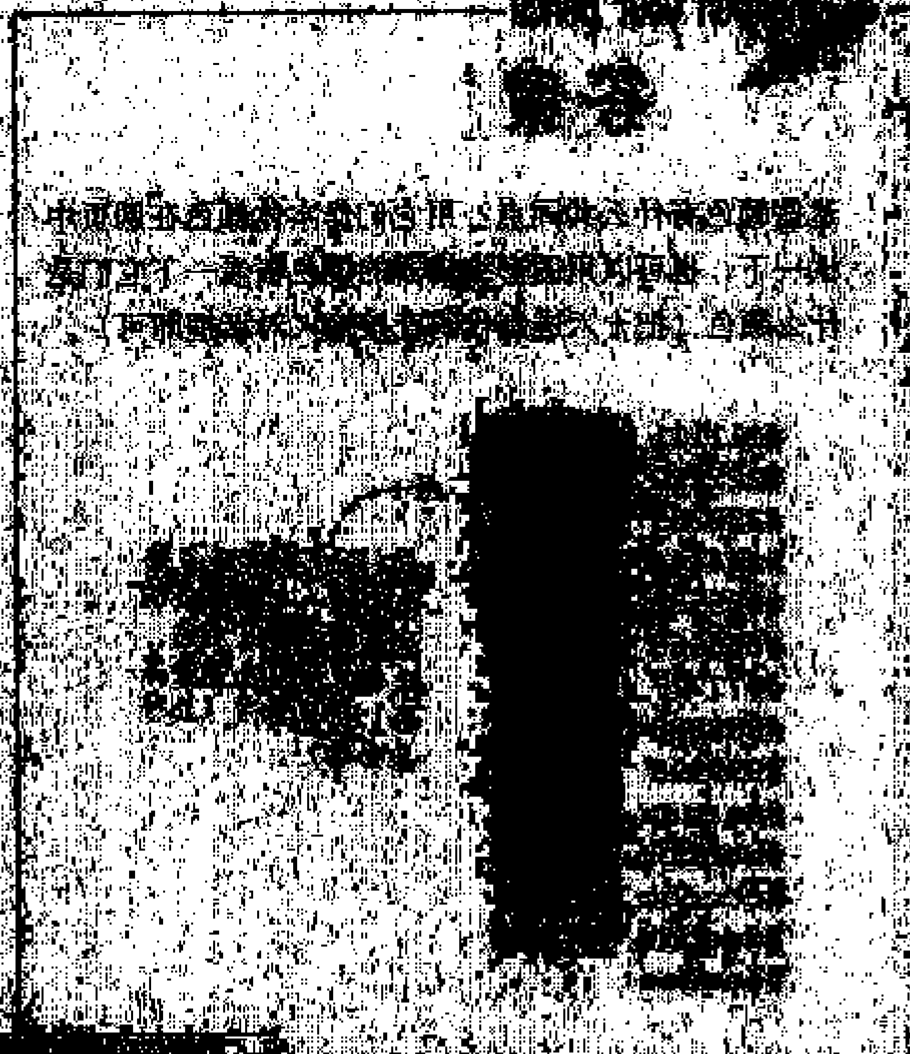
#111111
#222222
#333333
#444444
#555555
#666666
#777777
#888888
#999999
#aaaaaa
#bbbbbb
#cccccc
#dddddd
#eeeeee

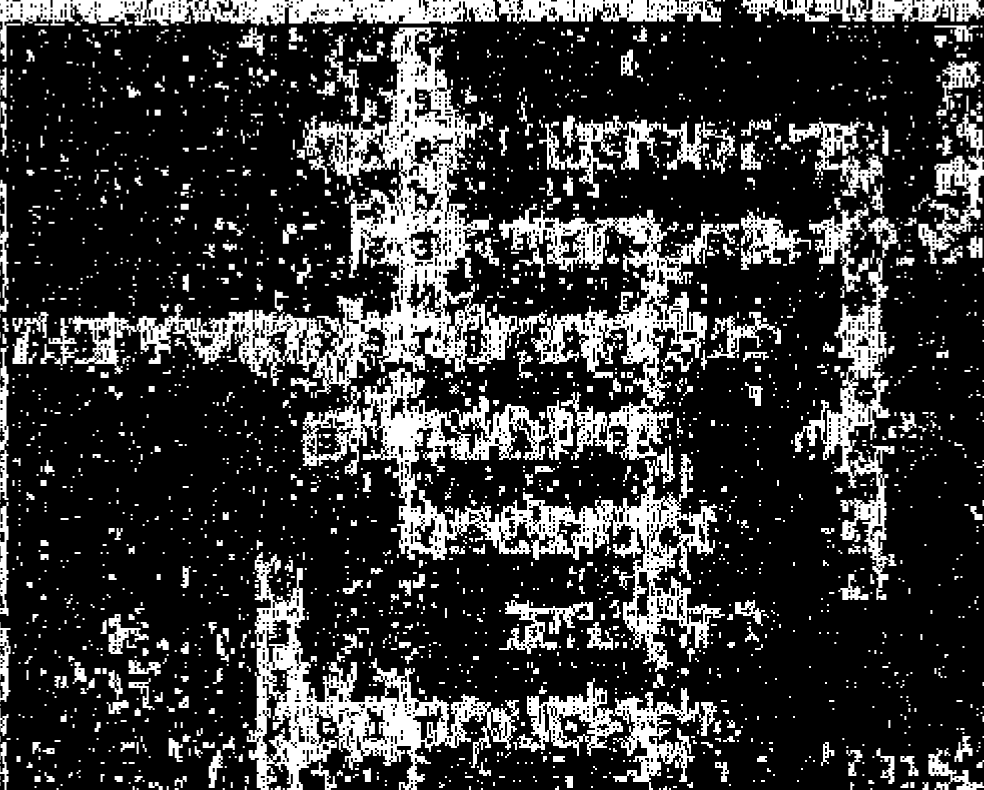


由十六进制代码全是相同的一个数组成的这些颜色都是灰色，从非常深（几乎是黑色）到非常浅（几乎是白色）。

XHTML填字游戏解答







10 盒模式

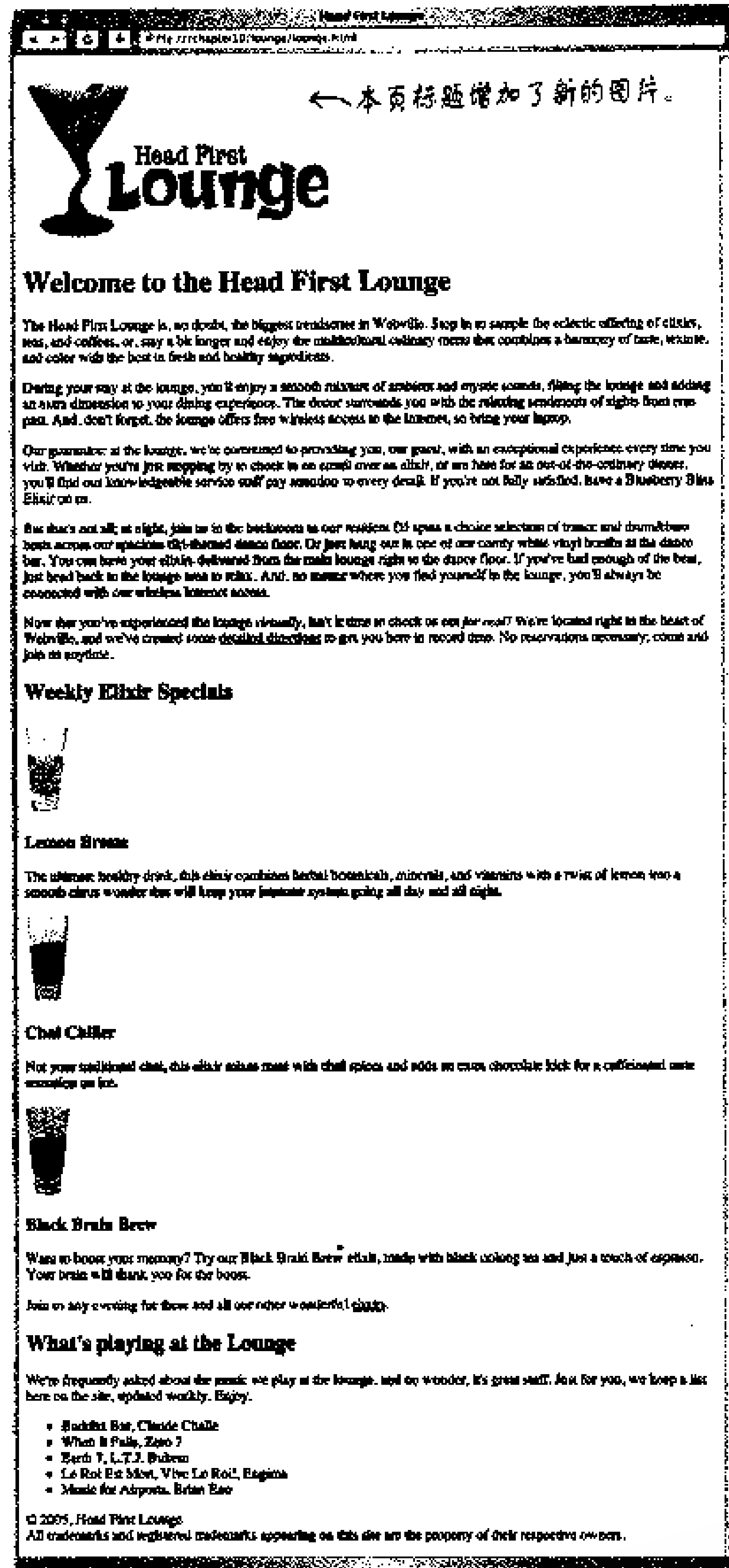
与元素亲密接触



要更好地构建网页你必须了解你的建筑材料。在这章中我们将仔细观察我们的建筑材料：XHTML元素。我们将仔细研究块和行元素，看看它们由什么组成。你将看到如何用CSS构建元素来控制各个方面。但我们不会止于此——你还会知道如何给元素指定唯一的标识符。如果这还不够，你还将学习什么时候及为什么要用多样式表。好的，翻开本页，让我们与元素来个亲密接触吧。

改进休闲室

你在前九章中学到了不少东西，Head First休闲室也经历了一系列的改进。在以后的两章中，我们将用全新的内容全面改进主页，从头开始样式化。为了激起你的好奇心，开始之前先来预览一下。请仔细观察——本页是换了新内容但没有样式化的新休闲室页面，下一页是下一章结束时我们会创建出来的样式化了的版本。

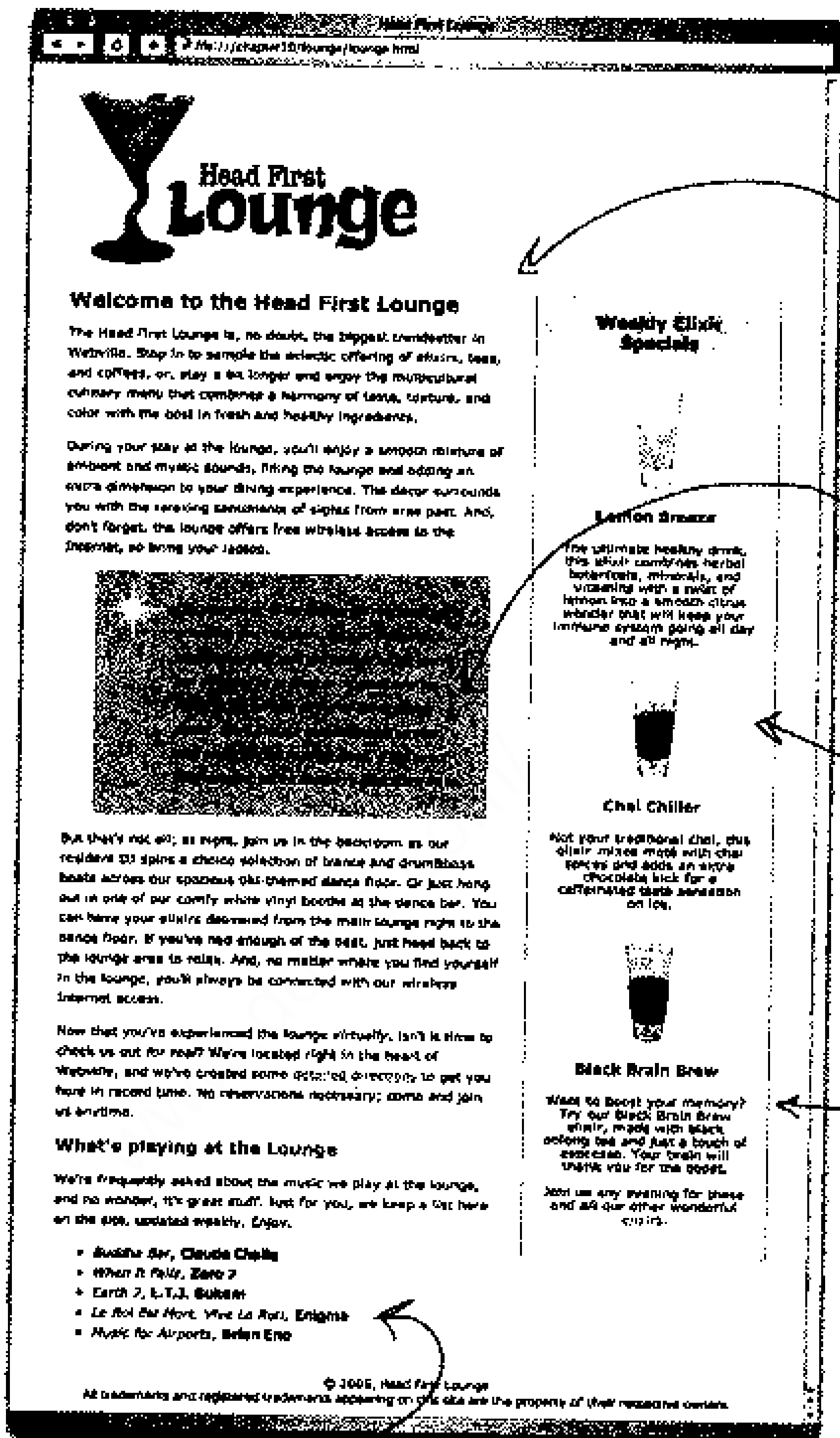


休闲室的伙计们提供了许多对休闲室和饮料的描述。

包括一系列专为本周提供的饮料。

他们还让顾客点播音乐，休闲室会在每个周末播放最热门的歌曲。

最后，在页脚写了一些版权声明。



改进过的，超漂亮的新休闲室

挺漂亮吧。休闲室设计对你来说好像有点过于“时尚”了吧，但要知道，这是个休闲室啊。你肯定觉得这个设计十分复杂——不过想想同样的技术能给你的网页带来什么效果呢。学完以下两章后，这样的设计对你来说就是小菜一碟了。

创建新休闲室

在开始构建之前，先熟悉一下新休闲室。你需要做的是：

1. 打开“chapter10/lounge”目录，你会发现内容全新的文件“lounge.html”。用编辑器打开文件浏览一下，所有内容看起来应该都很熟悉：标题，段落，一些图像，一个列表。

2. 本章的大部分时间都是给这个XHTML添加样式，所以需要一个地方来存放CSS。因为要在样式表文件“lounge.css”中给lounge创建全新的样式，所以你会发现在“lounge.html”文件里**<head>**内的**<link>**元素还在，但是样式表“lounge.css”以前的版本不见了。

```
<link type="text/css" rel="stylesheet" href="lounge.css" />
```

记住，这个**<link>**元素告诉浏览器去找一个叫做“lounge.css”的外部样式表。

3. 接着，需要在“chapter10/lounge”目录下创建新的“lounge.css”，这个文件将包括新休闲室的所有新CSS样式。

先做一些简单的改进

现在就开始给休闲室添加一些样式吧。给CSS添加一些新规则——用来作为改进休闲室的基础——如字体系列、大小和颜色（同时可以好好复习一下上一章的内容）。那么，打开你的“lounge.css”添加以下规则：

```
body {
        font-size:   small;
        font-family: Verdana, Helvetica, Arial, sans-serif;
}

h1, h2 {
        color: #007e7e;
}

h1 {
        font-size: 150%;
}

h2 {
        font-size: 130%;
}
```

这是页面的默认字体。

我们将把休闲室的font-family字体设置成sans-serif。挑选了一些可选的字体，并以一般字体sans-serif结束。

我们将把<h1>和<h2>元素的颜色设置为浅绿色，和logo中的杯子颜色保持一致。

给<h1>和<h2>设置合理的标题字体大小。因为我们要把它们设置成不同的大小，所以需要两个分开的规则而不能合并在一起。

快速测试

我们快速测试一下，看看这些样式有什么效果。务必完成所有的改动，然后保存并测试。

标题现在是sans-serif字体和与logo相匹配的颜色，为页面创造了一个主题。

段落文本也是sans-serif字体，因为每个元素都继承了<body>的font-family属性。

<h2>标题也有了新的颜色和sans-serif字体，而且稍微改小了一些。

没有给<h3>应用任何样式，所以它只是继承了<body>的font-family属性。



这个链接以默认的蓝色显示，看起来太突兀了，我们得调整一下（稍后）。

再次调整

在我们开始做一些更大的改动前，还需要稍微调整一下休闲室，这次调整要用到你以前没见过的一个新属性，但是在这点上你已经有了足够的经验，不需要我们做太多的提示。那就赶紧动手试试吧。

我们要做的是：调整整个页面中文本的行间距，这样可以增加行间的垂直距离。为此我们在**body**规则中加一条**line-height**属性：

增加文本的行间距可以提高可读性，还可以增加页面不同部分之间的对比度稍后你将看到它是如何工作的。

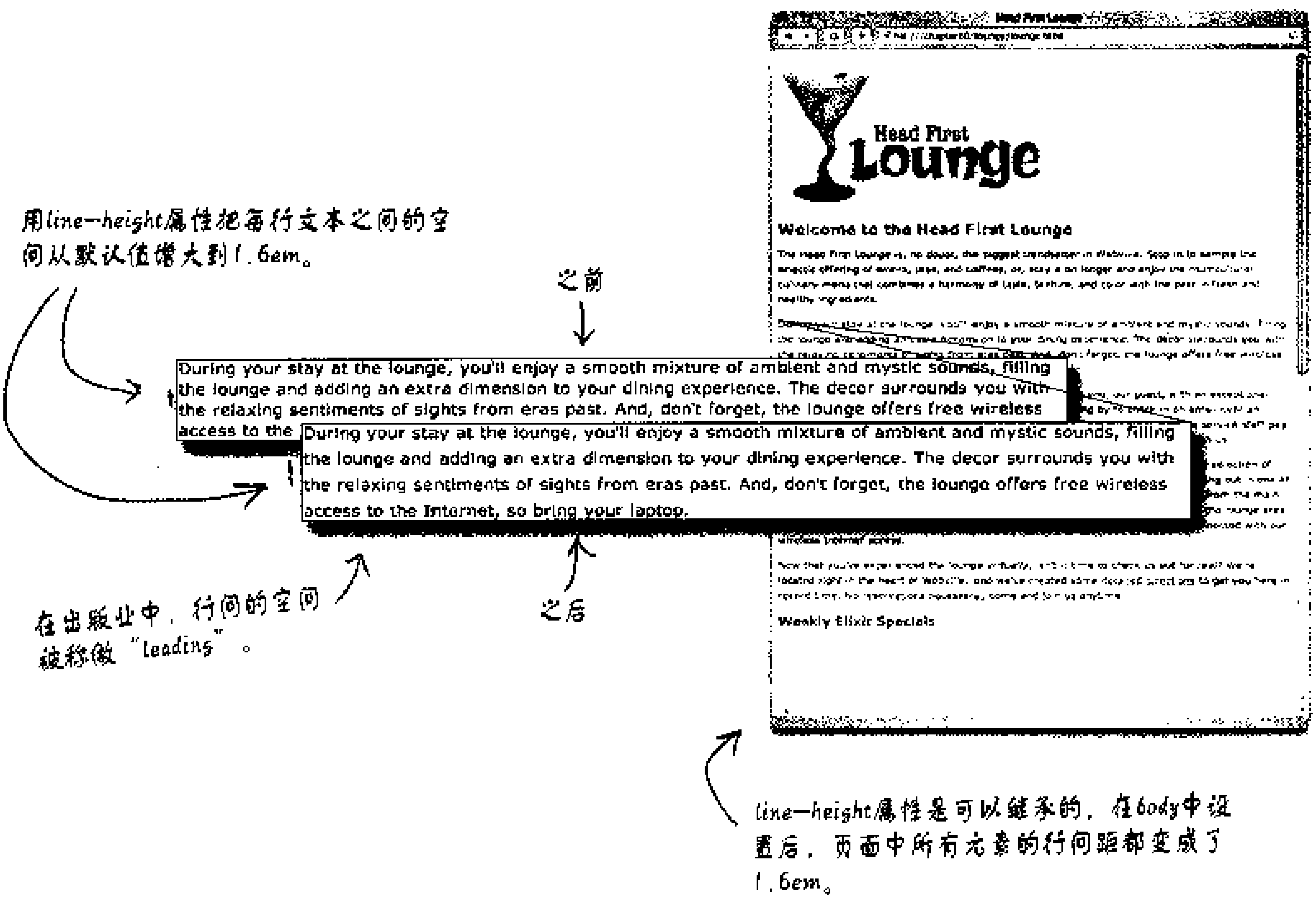
```
body {
      font-size:    small;
      font-family:  Verdana, Helvetica, Arial, sans-serif;
      line-height:  1.6em;
}
```

这里我们把行间距改为1.6em，相当于字体大小的1.6倍。

仔细观察新行间距

或许你已经猜到，使用line-height属性可以定义文本的行间距。跟其他与字体有关的属性一样，也可以用像素、em或百分数等与字体大小有关的值定义行间距。

看一下**line-height**属性对休闲室有什么影响。把**line-height**属性加入CSS文件并保存，刷新时你会发现行间距增大了。



练习

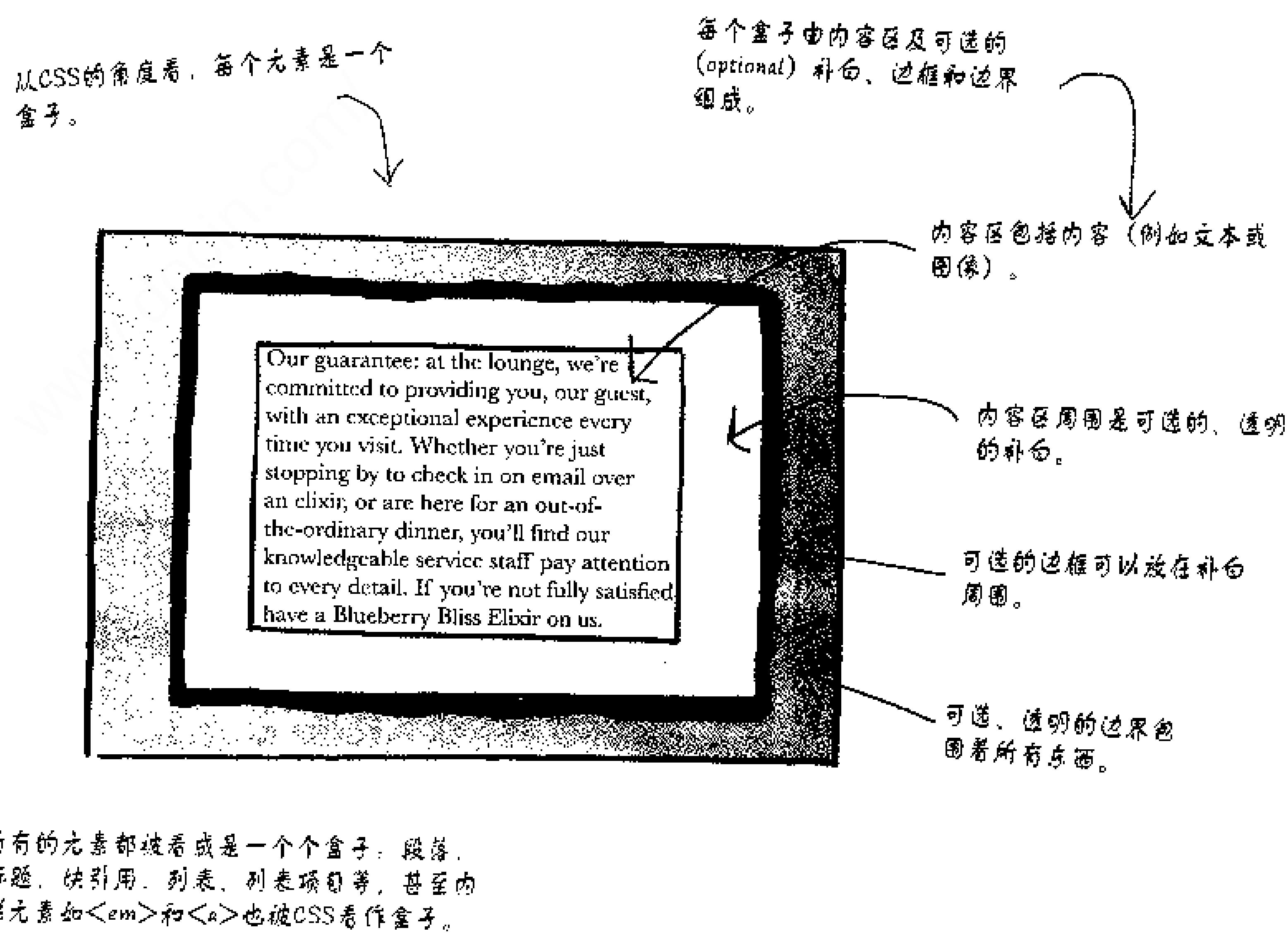
给line-height换几个不同的值试试，像200%、.5em和20像素来查看效果并比较哪个最好，哪个最差，哪个最具可读性。完成上述操作要确保将行间距改回原来的1.6em。

准备大翻新

学完这几页，新的休闲室已经有了许多文本样式，祝贺你！

现在一切将变得真正有趣，我们将不再只是改变一些简单的字体大小、颜色和修饰之类的元素属性，而是真正涉及到一些元素显示的基本原理。这是提高水平的开始。

为了上升到更高水平，你必须知道盒模式（the box model）。那是什么？CSS就是这样看待元素的，它把每个单一的元素看作是一个盒子。我们看一下这是什么意思。



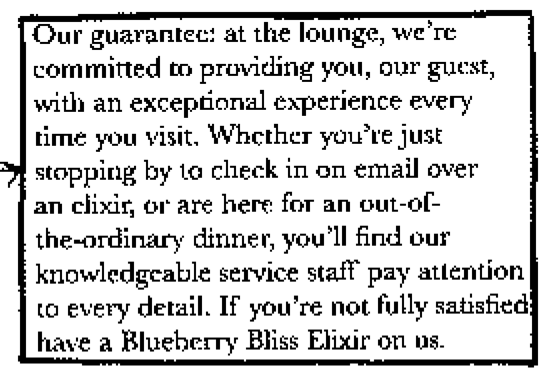
进一步解析盒模式（box model）

用CSS可以控制盒子的各个方面：内容周围补白的大小、元素有没有边框（以及类型和大小），以及元素之间有多少边界。我们仔细研究一下盒子的每一部分和它们的作用：

内容区包含元素的内容，其大小正好足够包含内容。

内容区是指什么？

每个元素都以某些内容开始，比如文本或图像，这部分内容被放置在一个大小正好能包含它的盒子里。注意内容区的内容和盒子边缘之间没有空白。

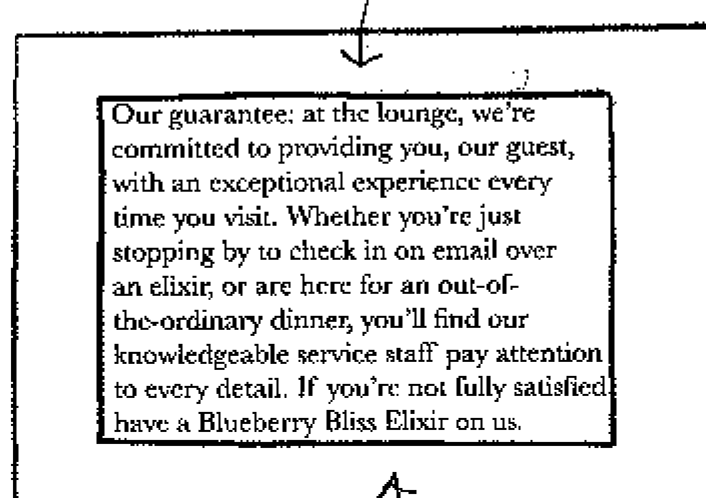


我们在内容区周围画一个边缘，只是为了让你知道它有多大，在浏览器中内容区周围并没有可见的边缘。

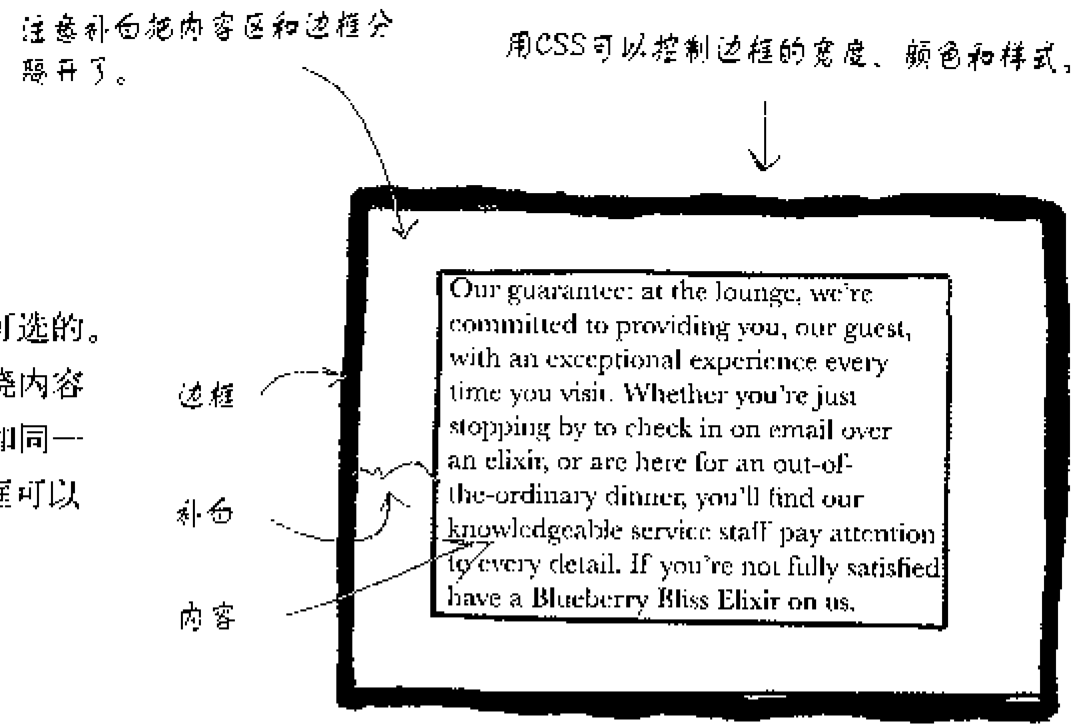
浏览器在内容区周围添加可选的补白。

补白是指什么？

任何盒子在内容区周围都能有一层补白。补白是可选的，所以不一定会有。可以用补白在盒子的内容和边框之间创建可视的空白。补白是透明的，本身没有颜色或修饰。

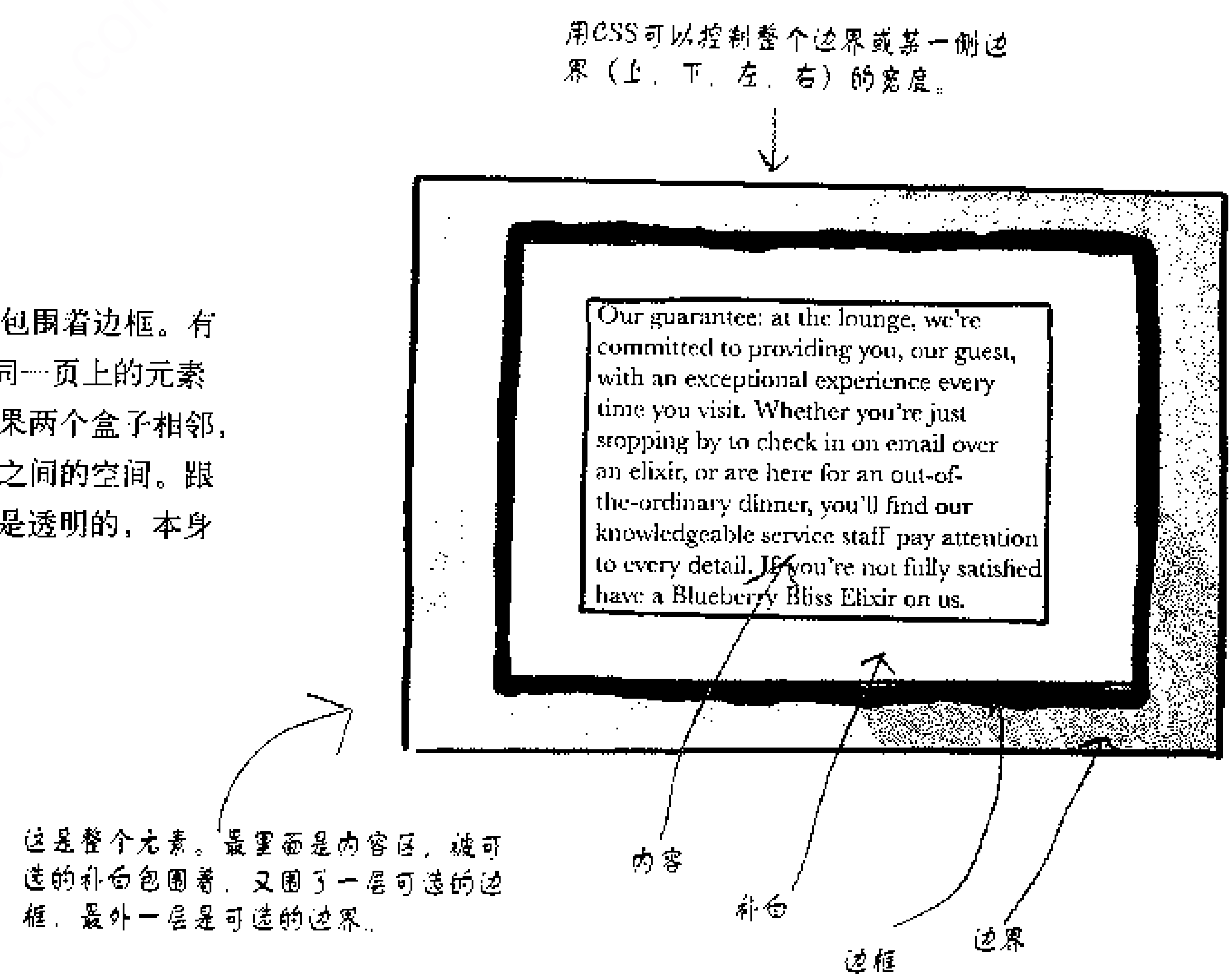


用CSS可以控制整个内容区周围甚至任何一侧（上，下，左，右）的补白宽度。



边框是指什么？

元素周围可以有边框，并且是可选的。边框包围着补白，因为它是围绕内容的一条线，所以视觉上将内容和同一页上的其他元素分隔开了。边框可以有各种宽度、颜色和样式。

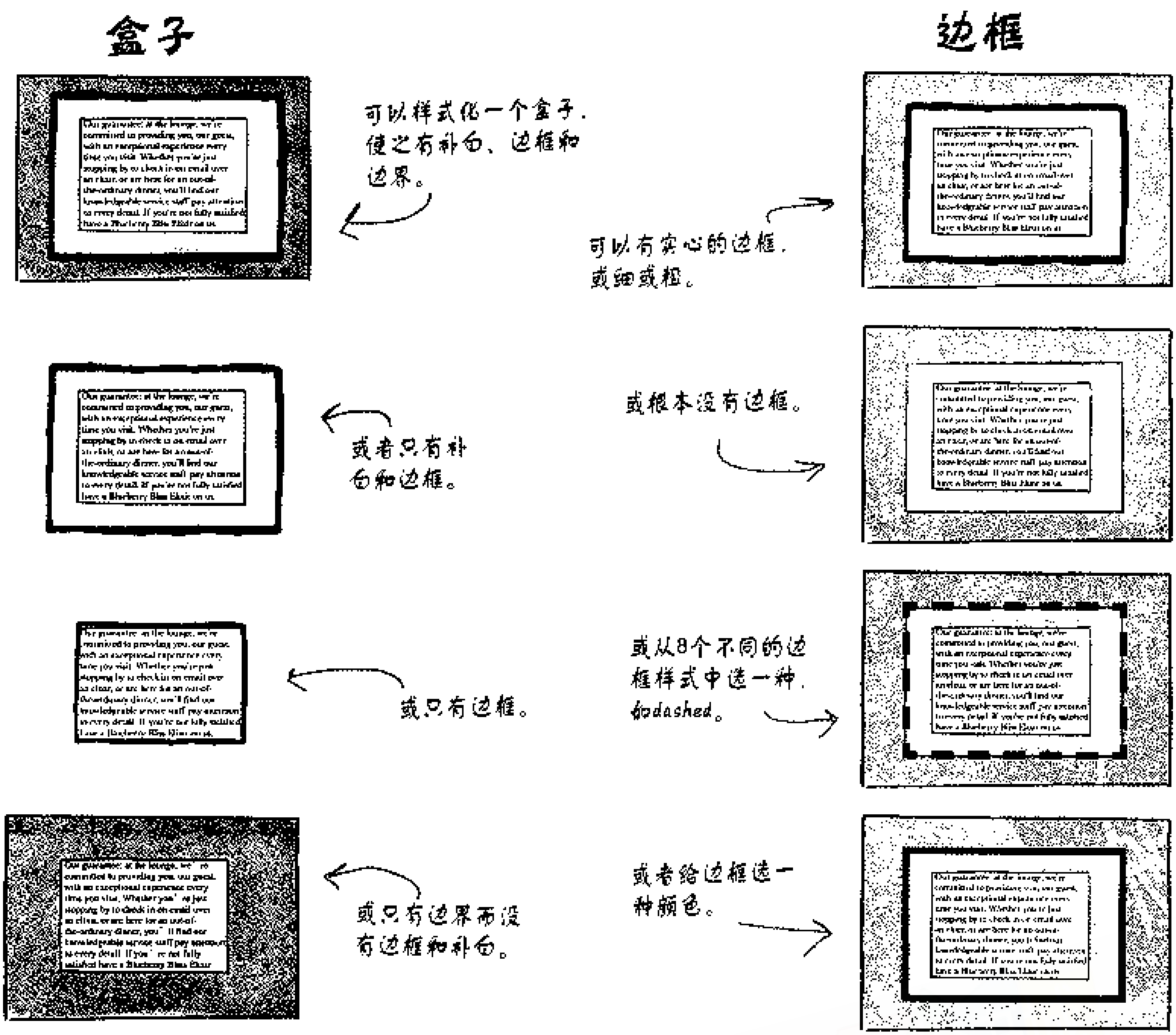


边界是指什么？

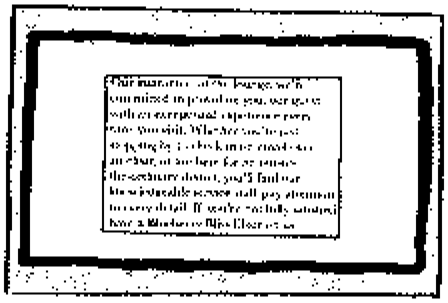
边界也是可选的，包围着边框。有了边界，就可以在同一页上的元素之间添加空间。如果两个盒子相邻，边界就相当于它们之间的空间。跟补白一样，边界也是透明的，本身没有颜色或修饰。

你能对盒子做什么？

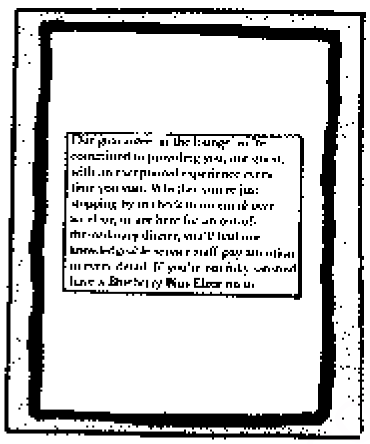
如果只有内容、补白、边框和边界，盒模式看起来也许比较简单。但是如果把这些全都结合起来，通过内部空间（补白）和周围空间（边界）来决定一个元素的布局的方法就有无穷多个。看几个变化元素方法的例子。



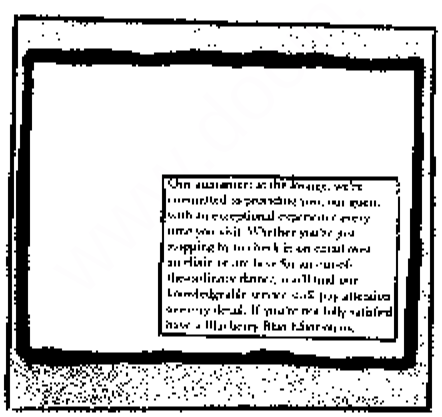
补白



用CSS可以把补白控制在内容区的任意一侧。这里左右补白要多一些。



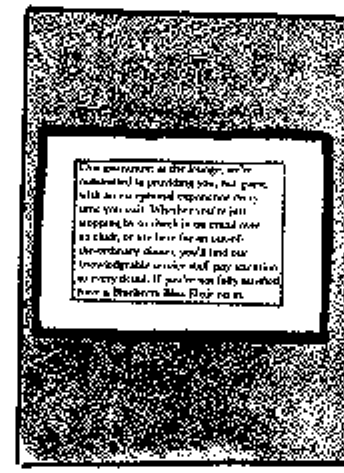
这里上下补白多一些。



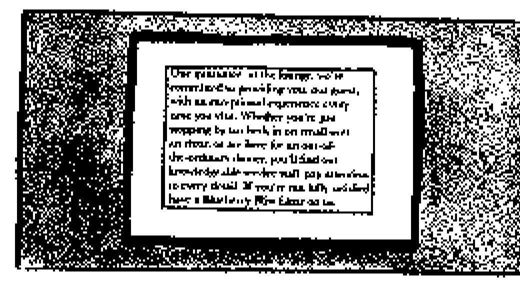
这里内容偏移到右下方，补白在左边和上边。

边界

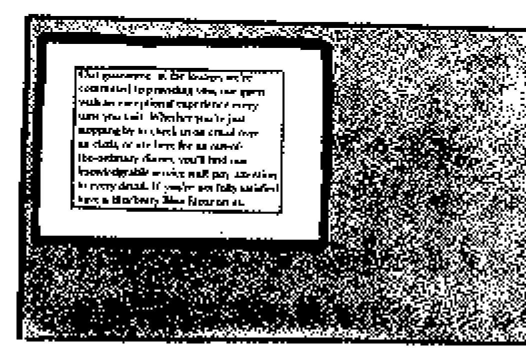
也可以同样地控制边界。这里上下边界多一些。



这里左右边界多一些。

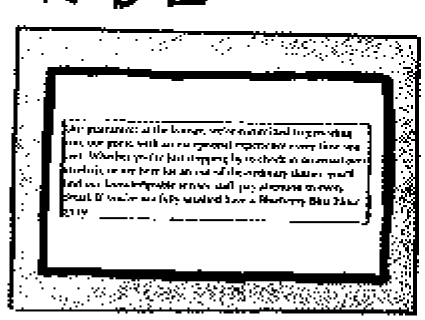


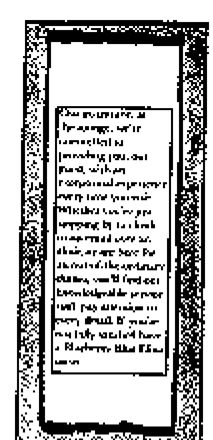
跟补白一样，也可以像这样单独地定义任何一侧的边界。



内容区

甚至可以用许多不同的方法控制高度和宽度。这个例子中，内容区比较宽。





这儿，内容区高而窄。

there are no Dumb Questions

问：如果我是给网页浏览器制作软件的，就会知道盒子这种东西很重要，但它能帮我制作更好看的网页吗？

答：为了不再使用浏览器默认的布局，你必须能够控制元素在页面上的位置，以及跟其他元素的相对位置。为此，要改变每个元素的补白和边界的各个方面。所以为了设计出吸引人的网页，绝对需要知道一些有关盒模式的知识。

问：补白和边界有什么区别？它们好像是一回事。

答：边界是元素之间的空间，而补白是内容周围多出来的空间。如果有可见的边框，补白在边框里面而边界在边框外面。把补白想成是元素的一部分，而边界包围着元素并把它和周围的内容隔离开。

问：我知道它们都是可选的，那如果想有边框或边界要不要先有补白？

答：不，它们都是可选的，互相没有依赖关系。所以可以有边框而没有补白，或有边界没边框等。

问：我不知道该如何安排元素，以及如何插入边界。

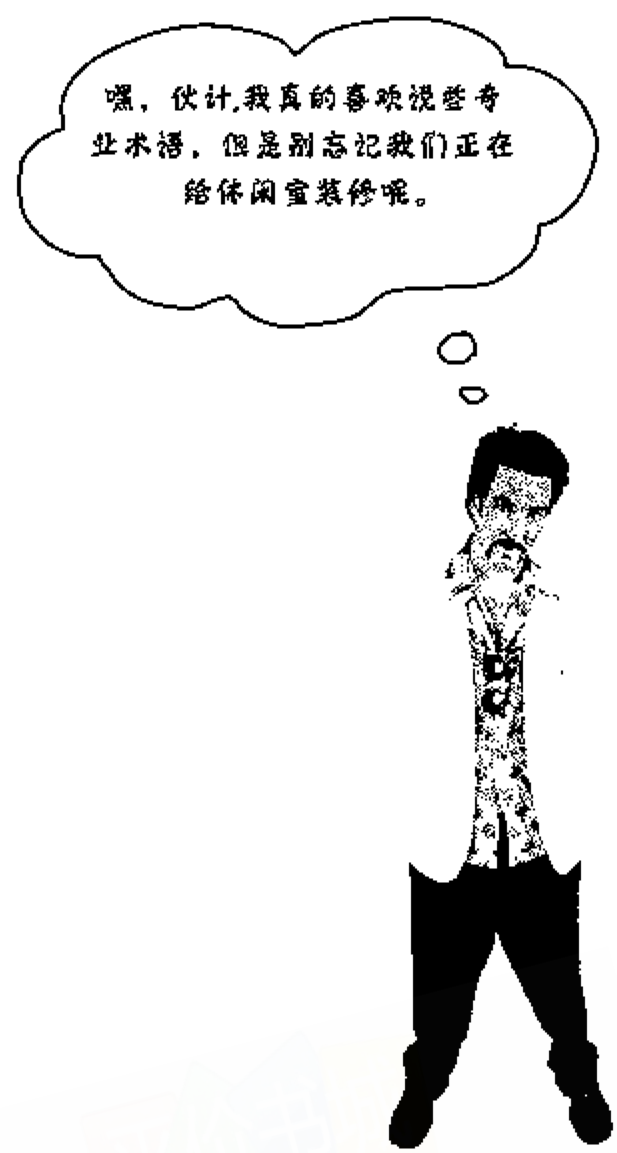
答：留着这个问题吧。你将在本章中学习一点边界是如何与别的元素合作的，然后会在第11章中讲到排版时学到这个问题。

问：除了大小以外，我好像不能样式化补白和边界的其他方面？

答：基本正确。两者都用来提供更多可见的空间，不能直接给补白或边界添加颜色或别的修饰，但因为它们是透明的，就会呈现背景颜色或背景图像。在这一点上补白和边界之间的不同是元素的背景颜色（或背景图像）会延伸到补白底下，但不会到边界。不久你会看到这是怎么回事。

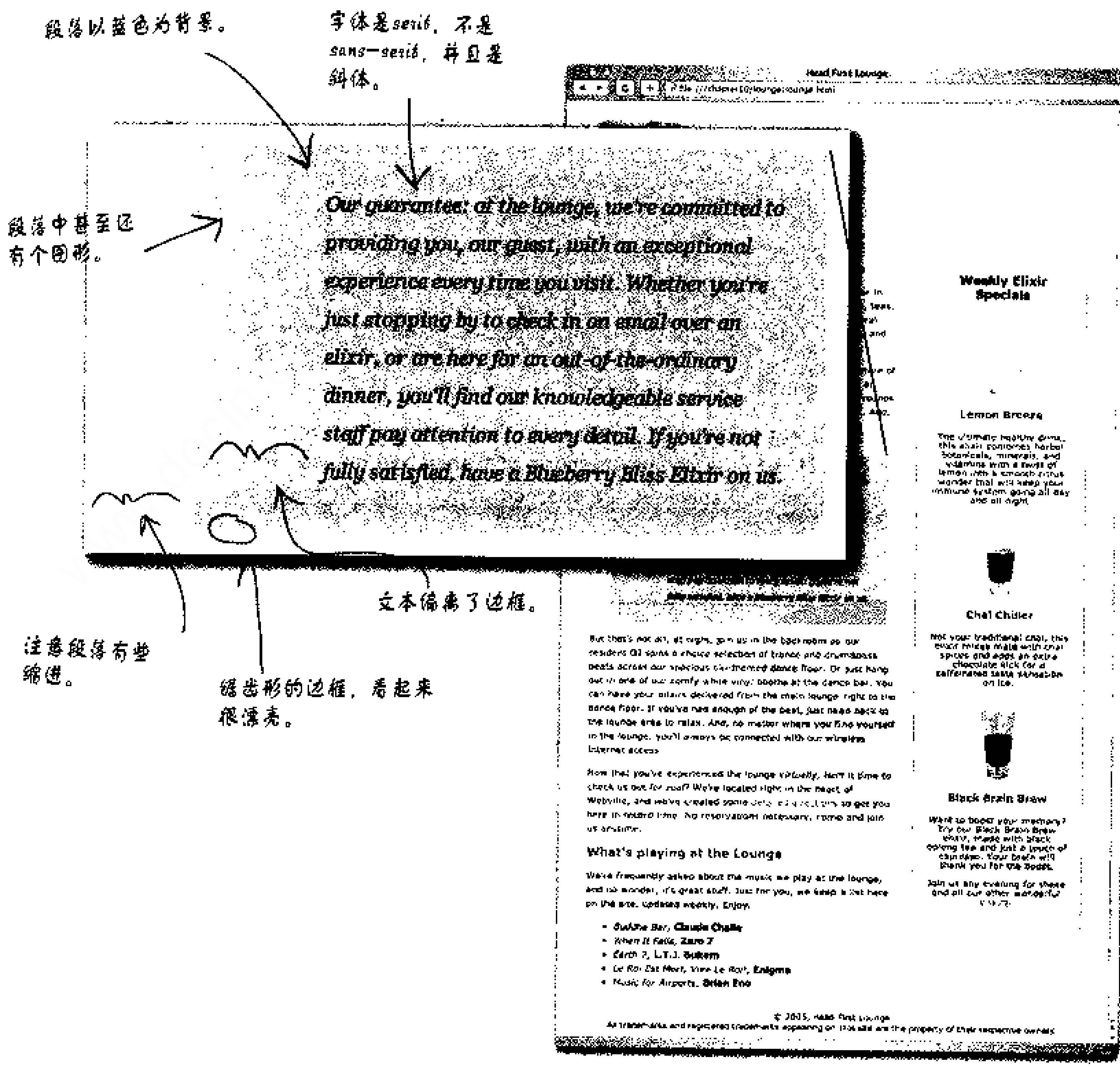
问：内容区的大小仅仅取决于其中内容的大小吗？

答：浏览器用许多不同的规则决定内容区的宽度和高度，以后我们会详述这方面的内容。简单来说，如果需要控制元素大小就可以自己设置宽度和高度。



返回休闲室……

我们已经有了在休闲室页面上继续工作的素材，那就回到页面去吧。你在本章开头看最新版的休闲室页面时，有没有注意蓝色的样式化了的那段？这段是休闲室对顾客的保证（guarantee），很显然他们想强调自己的承诺。我们仔细看看这段，然后动手把它做出来。



Sharpen your pencil

看看你能否区别这段的补白、边框和边界。标记出所有的补白和边界（上，下，左，右）：

don't forget, the lounge offers free wireless access to the Internet, so bring your laptop.

Our guarantee: at the lounge, we're committed to providing you, our guest, with an exceptional experience every time you visit. Whether you're just stopping by to check in on email over an elixir, or are here for an out-of-the-ordinary dinner, you'll find our knowledgeable service staff pay attention to every detail. If you're not fully satisfied, have a Blueberry Bliss Elixir on us.

But that's not all; at night, join us in the backroom as our resident DJ spins a choice selection of trance and drum&bass beats across our spacious tiki-themed dance floor. Or just hang

BRAIN POWER

在继续学习下一页之前，想一下如何用补白、边框和边界把普通段落变成“保证段落”。

创建保证样式

我们先稍微改变一下保证段落的样式，来熟悉一下段落的盒子是怎么形成的。为此要给段落添加一个叫做“guarantee”的类，这样就可以只给这段自定义样式了。再添加边框和背景颜色，这些会让你确切地明白段落是一个怎样的盒子。接着我们会学习其他的样式。你需要做的是：

1. 打开文件“lounge.html”并找到以“Our guarantee”开始的段落，给元素加一个叫做“guarantee”的类，如下所示：

添加类属性并设置其值为“guarantee”，记住，一个类让你能够单独给这个段落添加样式。

```
<p class="guarantee">
    Our guarantee: at the lounge, we're committed to providing
    you, our guest, with an exceptional experience every time you
    visit. Whether you're just stopping by to check in on email
    over an elixir, or are here for an out-of-the-ordinary dinner,
    you'll find our knowledgeable service staff pay attention to every
    detail. If you're not fully satisfied have a Blueberry Bliss Elixir
    on us.
</p>
```

2. 保存文件“lounge.html”并打开“lounge.css”。你将给保证段添加边框和背景颜色。把以下的CSS添加进你样式表的末尾并保存。

前3个属性给guarantee类里的所有元素添加边框，目前为止只是给这段添加。

```
.guarantee {
    border-color:      black;
    border-width:      1px;
    border-style:      solid;
    background-color:  #a7cece;
}
```

设置边框为黑色……

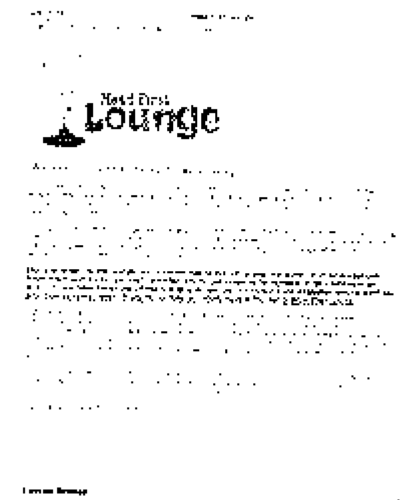
……宽度为1像素……

……实线。

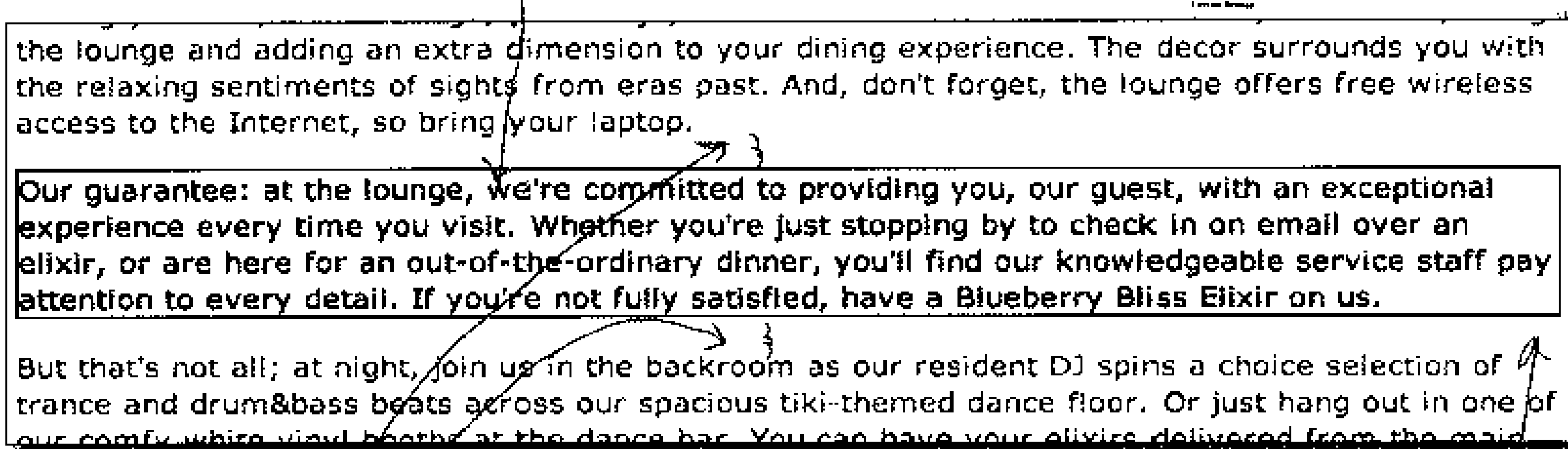
还给元素添加了背景颜色，让你可以看到补白和边界之间的不同，并使保证段显得好看一些。

段落边框测试

重新将页面载入浏览器，你会看到保证段落现在有了浅绿色的背景颜色，周围还有很窄的黑色边框。我们来仔细观察一下……



看起来好像段落内容周围没有补白——文本和边框之间没有空间。

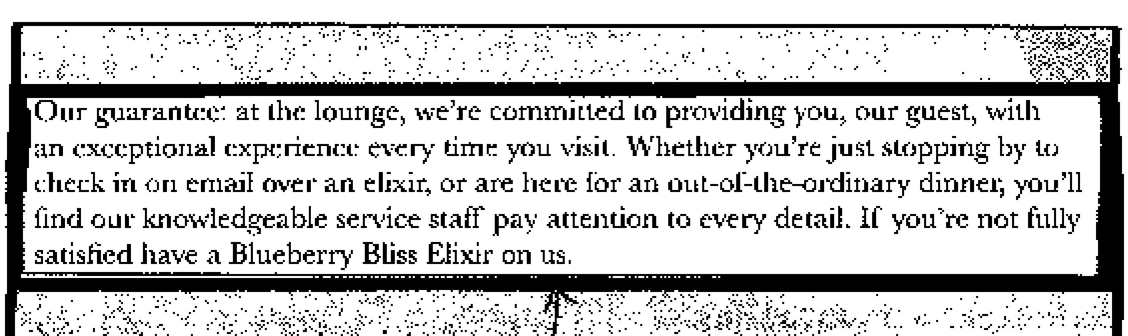


但是段落元素的顶部和底部似乎确实有边界。

段落的左右边缘和浏览器窗口边缘之间没有明显的边界。

如果我们把它画成盒模式示意图，这段就如下图所示：

顶部和底部有边界。

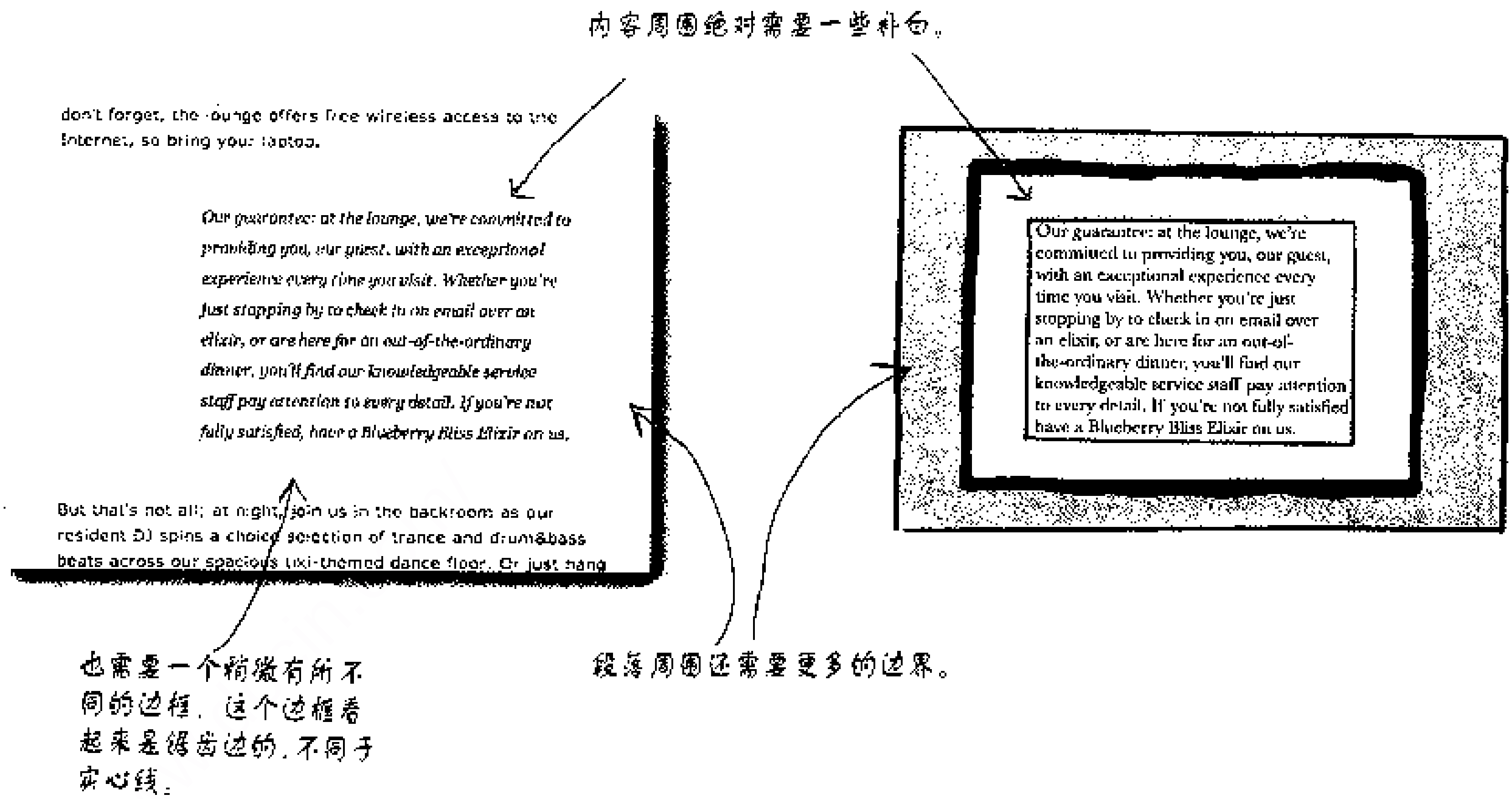


但左右边界很小。

边框紧挨着内容，也就是说补白设置得很小，或者说根本没有补白。

设置保证段的补白、边框和边界

既然你已经知道如何设置保证段落的补白、边框和边界了，那么想一下我们真正想要的样式。



添加一些补白

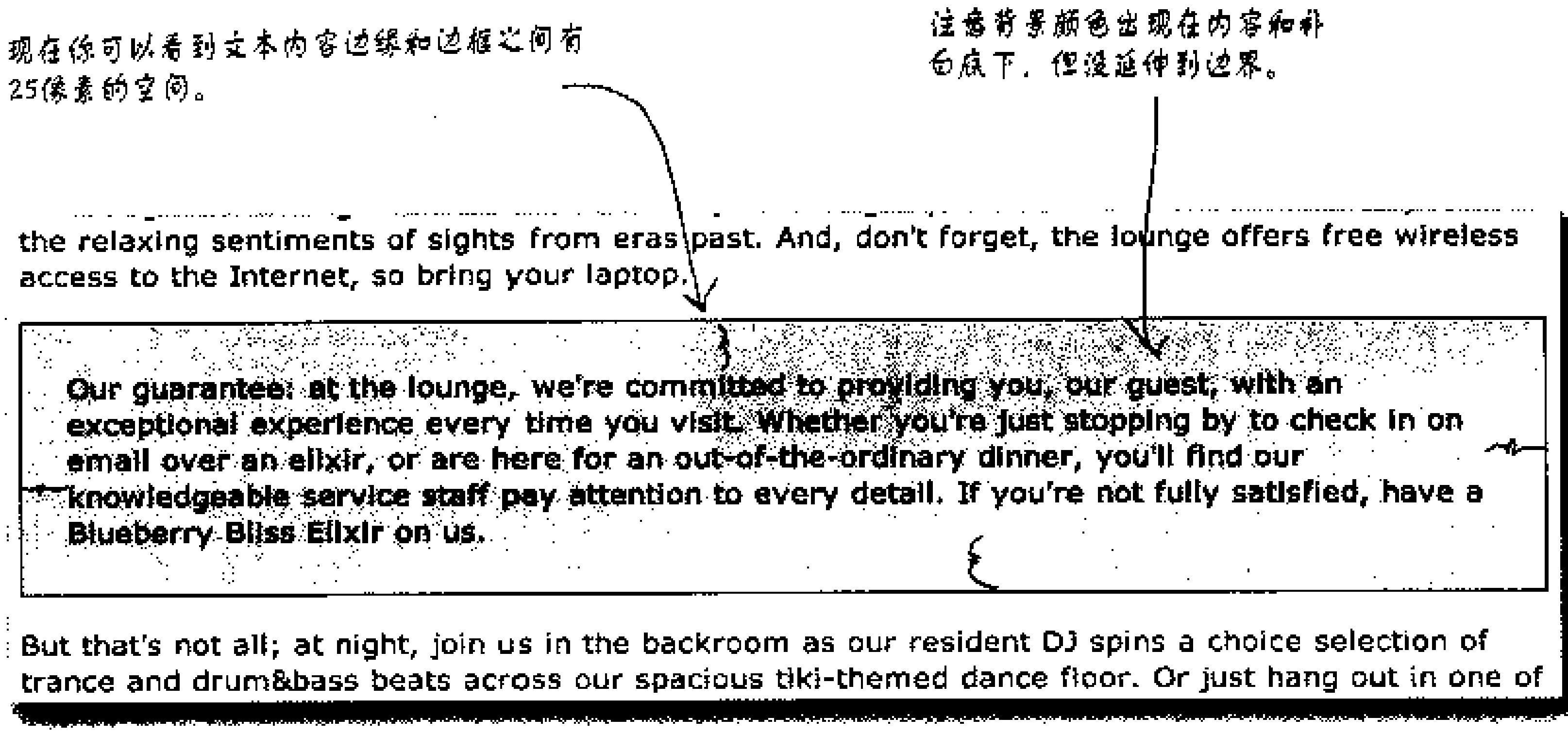
我们先设置补白。CSS有一个**padding**属性，可以用它在内容周围设置相同大小的补白。可以把这个属性设置成多少像素或边框里面部分的百分之几。我们把补白设置成25像素。

```
.guarantee {
        border-color:       black;
        border-width:       1px;
        border-style:       solid;
        background-color:   #a7cece;
        padding:            25px;
}
```

我们在内容的四周（上、下、左、右）各添加了25像素的补白。

测试补白

把页面重新载入浏览器后，你会发现保证段落的文本周围（文本和边框之间）多了一些呼吸空间，易读多了。



添加一些边界

用CSS添加边界很容易，和补白一样，可以用百分数或像素定义边界。要在整个保证段落周围添加30像素的边界，可以这么做：

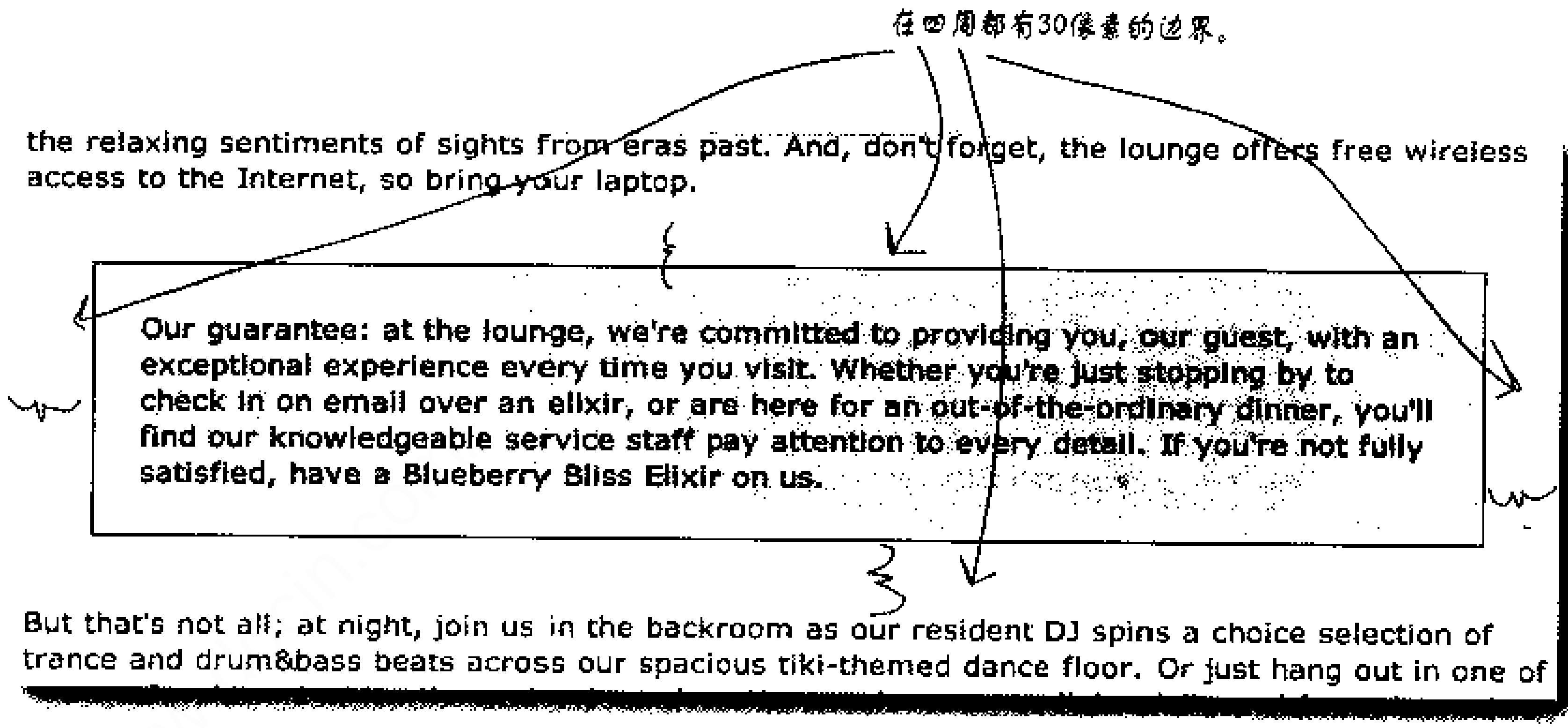
```
.guarantee {
      border-color:      black;
      border-width:      1px;
      border-style:      solid;
      background-color:  #a7cece;
      padding:           25px;
      margin:            30px;

}
```

我们在内容四周（上、下、左、右）各添加了30像素的边界。

测试边界

重新加载休闲室页面，你会发现保证段在网页中显得比较突出。边界使这个段落显得与众不同，再加上背景颜色，这段看起来有鹤立鸡群的效果。如你所见，只用几行CSS就做了一些非常棒的事情。



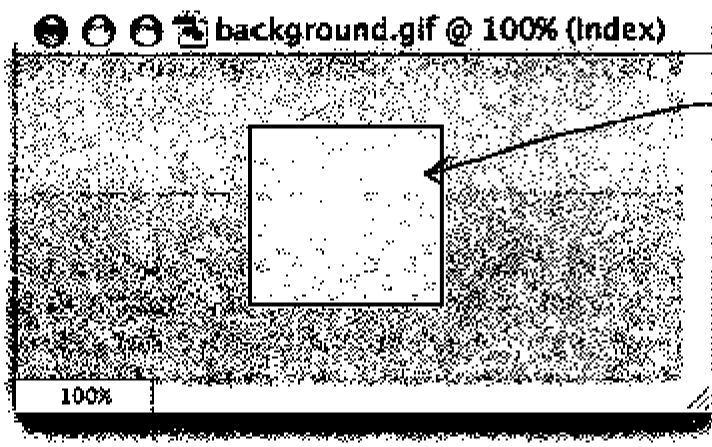
练习

看一下最后一种版本的保证段落，它的字体是斜体、serif，行间距比页面中其他文本的要大。如果再看仔细一些，文本是灰色的。按以下要求写CSS，把行间距设置为1.9em，字体样式为italic，颜色为#444444，字体列表为Georgia、"Times New Roman"、Times、serif。核对答案（在本章后），输入并测试。

插入背景图像

你快要完成了。还差什么呢？我们还需要将白色的图形“guarantee star”添加进段落，并且把黑色实心线条的边框修改一下。让我们先来处理图像。

看一下“chapter10/lounge/images”目录，你会找到一个名为“background.gif”的GIF图像，如下图所示：



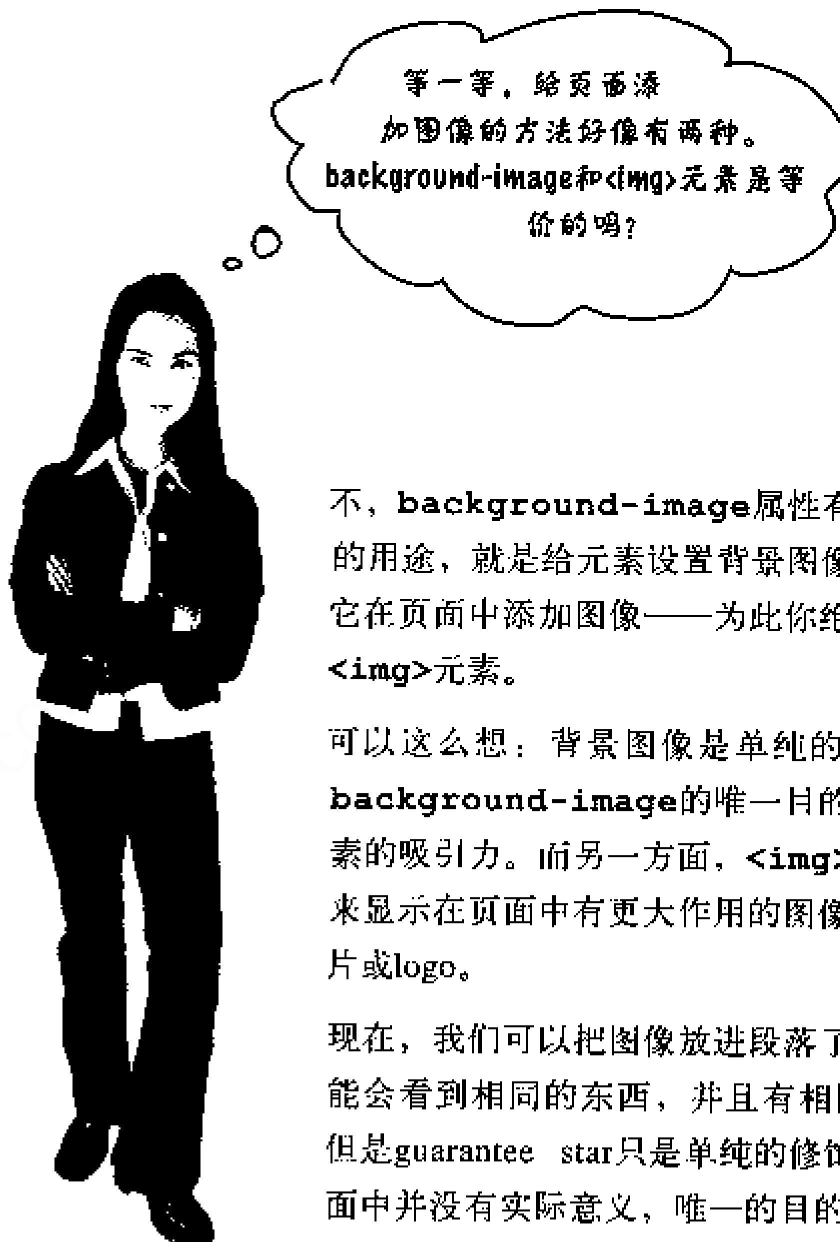
这个图像是透明背景上的一个像星星的简单白色图案。注意周围还有一些和背景颜色相匹配的色彩。

现在，只需把图像插入段落元素就可以了，所以你会用到**<img>**元素，对吗？不用这么着急，要添加一个图像作为一个元素的背景，还有另一种方法。用CSS，可以用**background-image**属性给任何元素添加背景图像。我们来试一下，并看看它是如何工作的：

这些是在上一页的练习中你添加的属性。

```
.guarantee {
        line-height:       1.9em;
        font-style:        italic;
        font-family:       Georgia, "Times New Roman", Times, serif;
        color:             #444444;
        border-color:      black;
        border-width:      1px;
        border-style:      solid;
        background-color:  #a7cece;
        padding:           25px;
        margin:            30px;
        background-image:  url(images/background.gif);
}
```

把这条规则添加到你的CSS中，保存并重新载入页面。



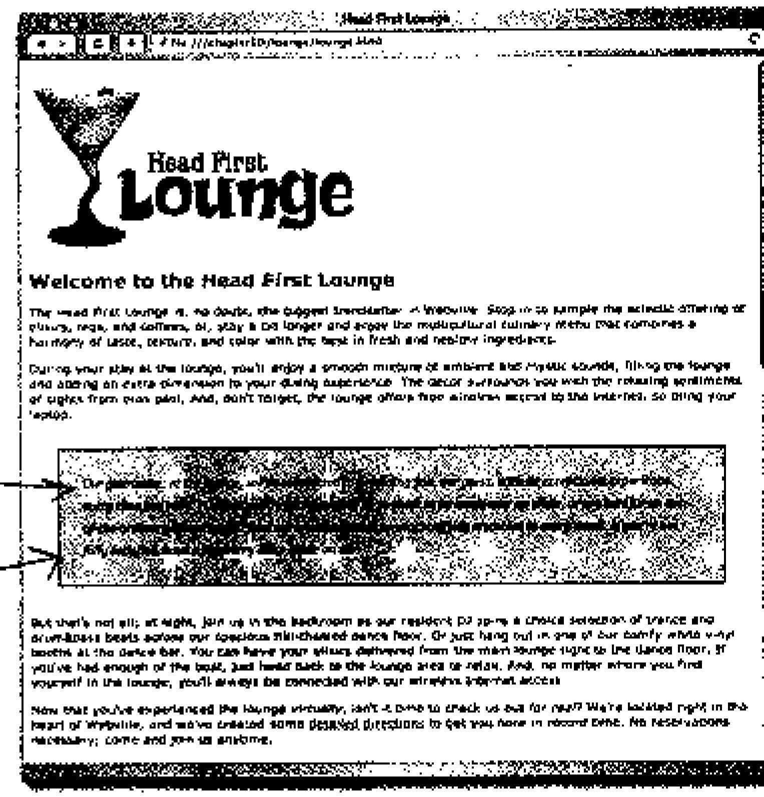
不，**background-image**属性有一个特定的用途，就是给元素设置背景图像。不能用它在页面中添加图像——为此你绝对需要用**<img>**元素。

可以这么想：背景图像是单纯的外观，用**background-image**的唯一目的是提高元素的吸引力。而另一方面，**<img>**元素，用来显示在页面中有更大作用的图像，比如照片或logo。

现在，我们可以把图像放进段落了，我们可能会看到相同的东西，并且有相同的感受，但是guarantee star只是单纯的修饰——在页面中并没有实际意义，唯一的目的是使元素更美观一些。所以**background-image**更多是用来制造视觉效果。

测试背景图像

这的确是个有趣的测试——背景图像出现了，但好像重复了许多次。我们来仔细看一下如何使用CSS背景图像，然后你就能搞定它了。



这是背景中的图像guarantee star。注意它是置于背景颜色之上的，并且因为它的背景是透明的，所以颜色不见了。

还要注意背景图像——跟背景颜色一样——只出现在内容区和补白之下，没有延伸到边框之外的边界。

再靠近一点CSS

除了用background-image属性可以给元素设置背景图像之外，还有两个属性也能影响背景图像：background-position和background-repeat。

```
background-image: url(images/background.gif);
```

background-image属性被设置成URL，可以是相对路径或完整的URL（http://…）。

注意URL没有加引号。

搞定背景图像

默认的背景图像是重复的。还好**background-repeat**属性有**no-repeat**值。浏览器默认的背景图像的位置是元素的左上方，这正是我们想要的，不过还是添加**background-position**属性来试试看吧。

```
.guarantee {
        line-height:           1.9em;
        font-style:            italic;
        font-family:           Georgia, "Times New Roman", Times, serif;
        color:                 #444444;
        border-color:          black;
        border-width:          1px;
        border-style:          solid;
        background-color:      #a7cece;
        padding:               25px;
        margin:                30px;
        background-image:      url(images/background.gif);
        background-repeat:     no-repeat;
        background-position:   top left;
}
```

添加了两个新属性。

我们想让背景图像不重复。

并想把它放在左上角。

background-repeat属性用来设置图像的位置，可以用像素、百分数或top，left，right，bottom，center这些关键字定义。

```
background-position: top left;
```

把图像放置在元素的左上方。

CSS中有许多确定事物位置的方法，在后两章中我们会讲到更多。

默认的背景图像是"tiled"，或者在背景上再三重复。

background-repeat属性用来控制这种平铺效果。

```
(background-repeat: repeat;)
```

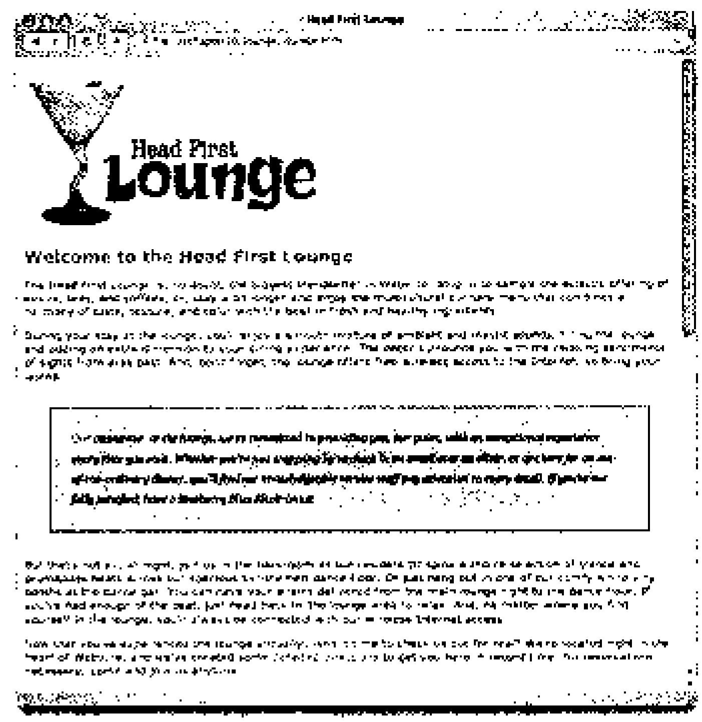
把图像设置为在水平和垂直方向上都重复，这是默认的格式。

这儿还有另外几种background-repeat值可以用。

- **no-repeat** ← 图像只显示一次，不重复。
- **repeat-x** ← 图像只在水平方向上重复。
- **repeat-y** ← 图像只在垂直方向上重复。
- **inherit** ← 继承父元素的值。

再一次测试背景图像

我们再来一次。这次，好像离我们想要的效果不远了。但因为这是个背景图像，文本会处于它之上。如何解决这个问题呢？这就得靠补白了！补白用来在内容区周围添加可见的空间。我们在左边增加补白来看一下能不能一次就全部搞定。



这样好一些，现在图像不重复了。

但我们实在不想让文本盖过图像。

如何只在左侧添加补白？

CSS在上，下，左，右每个方向都有补白、边界甚至边框属性。要在左侧添加补白，就要用padding-left属性，如下所示：

```
.guarantee {
        line-height:            1.9em;
        font-style:             italic;
        font-family:            Georgia, "Times New Roman", Times, serif;
        color:                  #444444;
        border-color:           black;
        border-width:           1px;
        border-style:           solid;
        background-color:       #a7cece;
        padding:                25px;
        padding-left:           80px;
        margin:                 30px;
        background-image:       url(images/background.gif);
        background-repeat:      no-repeat;
        background-position:    top left;
}
```

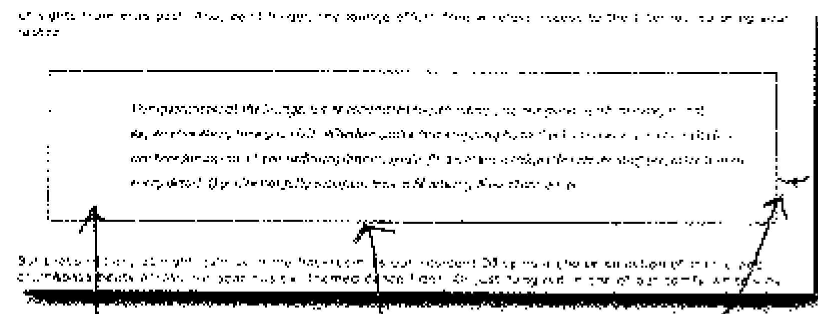
我们用padding-left属性来增加左侧的补白。

注意我们先把四周的补白都设置成25像素，然后再给左侧定义一个属性。

这儿顺序很重要——如果交换了顺序，就先设置了左侧的补白，然后总的补白属性就会把四周的补白全部设置成25像素，包括左侧的。

我们完成了吗？

保存设置并重新加载页面。你会看到段落左侧的补白增加了，文本跟guarantee star分开，位置很恰当。这是一个用补白代替边界的很好的例子。如果在内容区本身周围需要更多可见的空间，就用补白；相反地，如果想在元素之间或元素和页边之间有空间，就用边界。其实我们还可以在右边增加一些边界来使段落更加突出，我们先来做这个，然后剩下的就只有确定边框了。



补白看起来很不错。现在文本和图像分开了，位置很恰当。

还需要更好一些的边框。

可以增加右边的边界来使这段显得更突出一些。

如何只在右侧增加边界？

和增加补白的操作一样：添加另外一个属性**margin-right**，来增加右侧的边界。

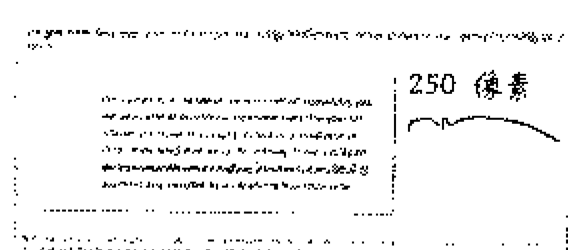
看到例子了吧：有一个可以一起控制四侧的边界的属性，还有可以单独控制各侧的边界的属性。

```
.guarantee {
        line-height:            1.9em;
        font-style:             italic;
        font-family:            Georgia, "Times New Roman", Times, serif;
        color:                  #444444;
        border-color:           black;
        border-width:           1px;
        border-style:           solid;
        background-color:       #a7cece;
        padding:                25px;
        padding-left:           80px;
        margin:                 30px;
        margin-right:           250px;
        background-image:       url(images/background.gif);
        background-repeat:      no-repeat;
        background-position:    top left;
}
```

记住我们已经把边界设置成了30像素。

现在我们将覆盖右侧的值，重新设置为250像素。

添加新的**margin-right**属性并重新载入。现在段落的右边将有250像素的边界。

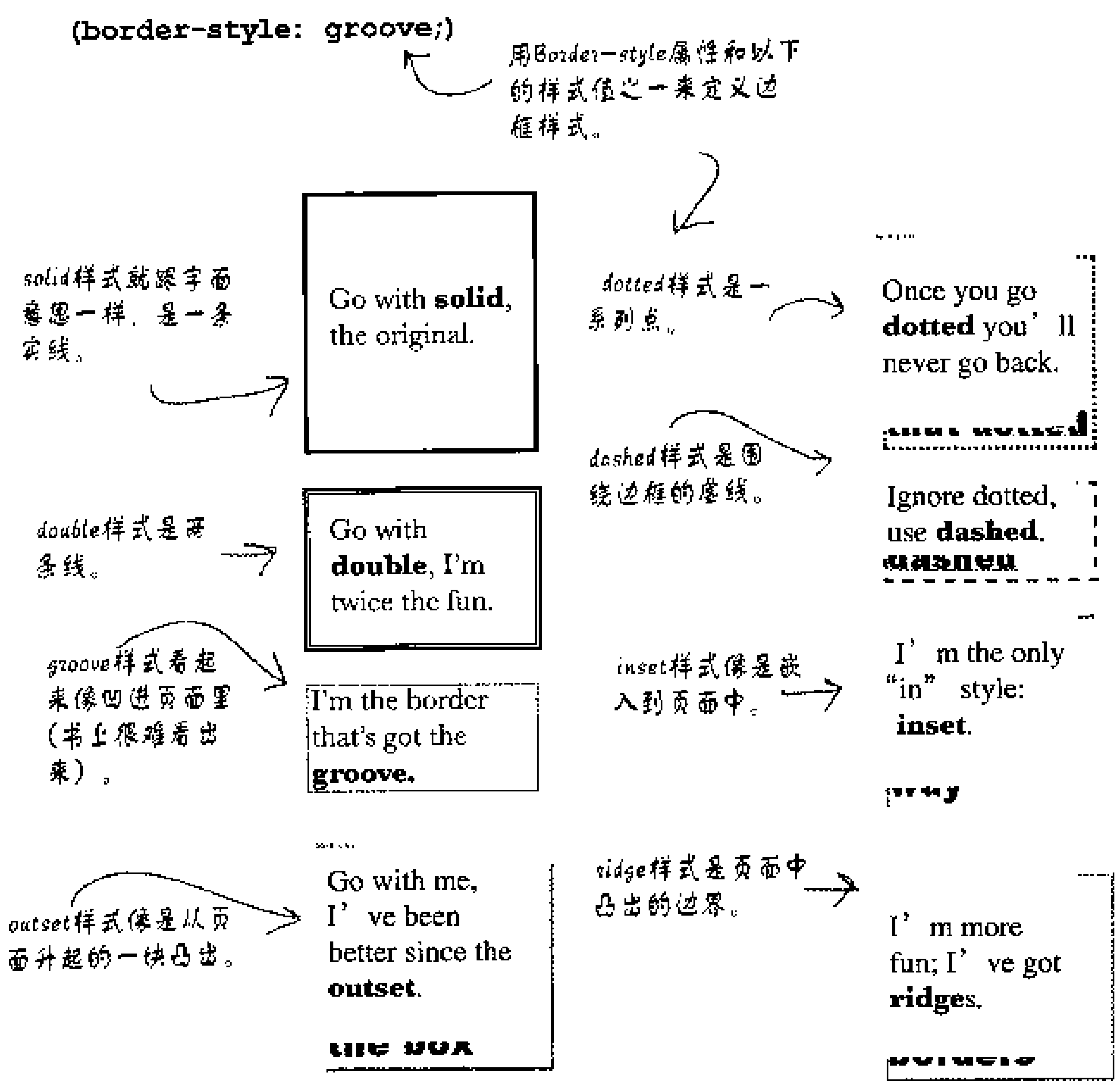


边框快速指南

现在只要再做一件事，做一个更漂亮的边框，保证段落就将变得完美了。动手之前，先看一下控制一个元素边框的所有不同方法。

边框样式

Border-style属性用来控制边框的外观。一共有8种边框样式，有实线、虚线、凸出线、凹进线等。

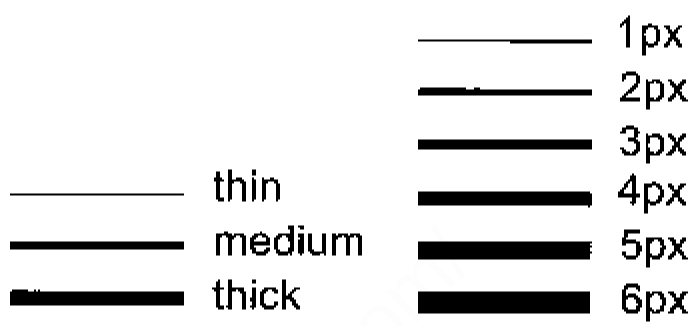


边框宽度

Border-width属性用来控制边框宽度。可以用关键词或像素来定义宽度。

```
border-width: thin;
border-width: 5px;
```

可以用关键字thin，medium，thick或像素值定义宽度。

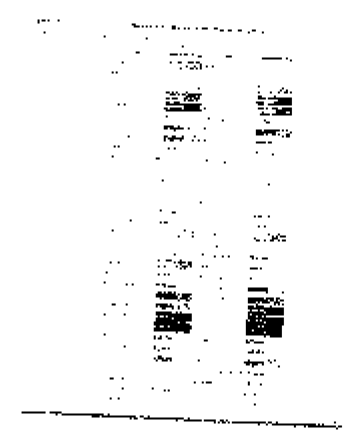


边框颜色

Borde-color属性用来设置边框颜色。和设置字体颜色一样，可以用颜色名称、rgb值或十六进制代码定义颜色。

```
border-color: red;
border-color: rgb(100%, 0%, 0%);
border-color: #ff0000;
```

用borde-color属性来定义边框颜色。可以用任何一段的方法定义颜色。



定义边框的一侧

border-top-color
border-top-style
border-top-width

border-right-color
border-right-style
border-right-width

border-bottom-color
border-bottom-style
border-bottom-width

border-left-color
border-left-style
border-left-width

跟边界和补白一样，也可以定义任何一侧（上，下，左，右）的边框样式、宽度或颜色。

```
border-top-color: black;
border-top-style: dashed;
border-top-width: thick;
```

这些属性只定义了上边框。可以单独地定义每侧的边框。

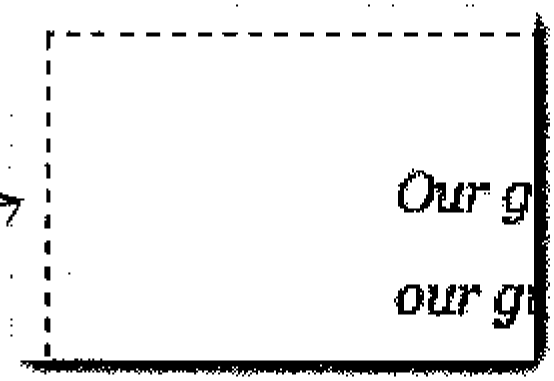
确定边框并完成

保证段落就要完成了。现在我们需要做的只是让边框看起来有锯齿边效果。那是什么样式呢？现有的样式是solid、dotted、double、dashed、groove、ridge、inset和outset。那我们怎么使它看起来有锯齿呢？其实不过是个小窍门：用虚线边框，把它颜色设置成白色（跟页面的背景颜色匹配）。下面就教你怎么做。先来设置虚线边框，在“lounge.css”中找到**border-style**属性并修改它，像这样：

```
border-style: dashed;
```

在这里我们把边框从实线改为虚线。

继续，保存文件并重新载入，你会看到这样一个边框：

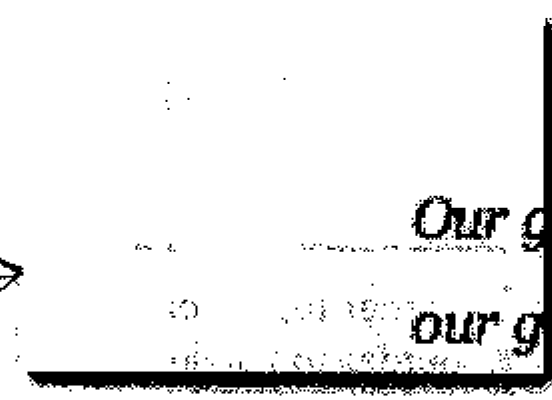


现在，为了得到看上去有锯齿的边框，只需要把边框颜色设置为白色就可以了。这样看起来好像背景颜色侵入了边框一样。试一下：找到**border-corlor** 属性并将其设置为**white**。

```
border-color: white;
```

把边框颜色从黑色变成了白色。

再次保存文件并重新载入，现在可以看到锯齿边的边框了：

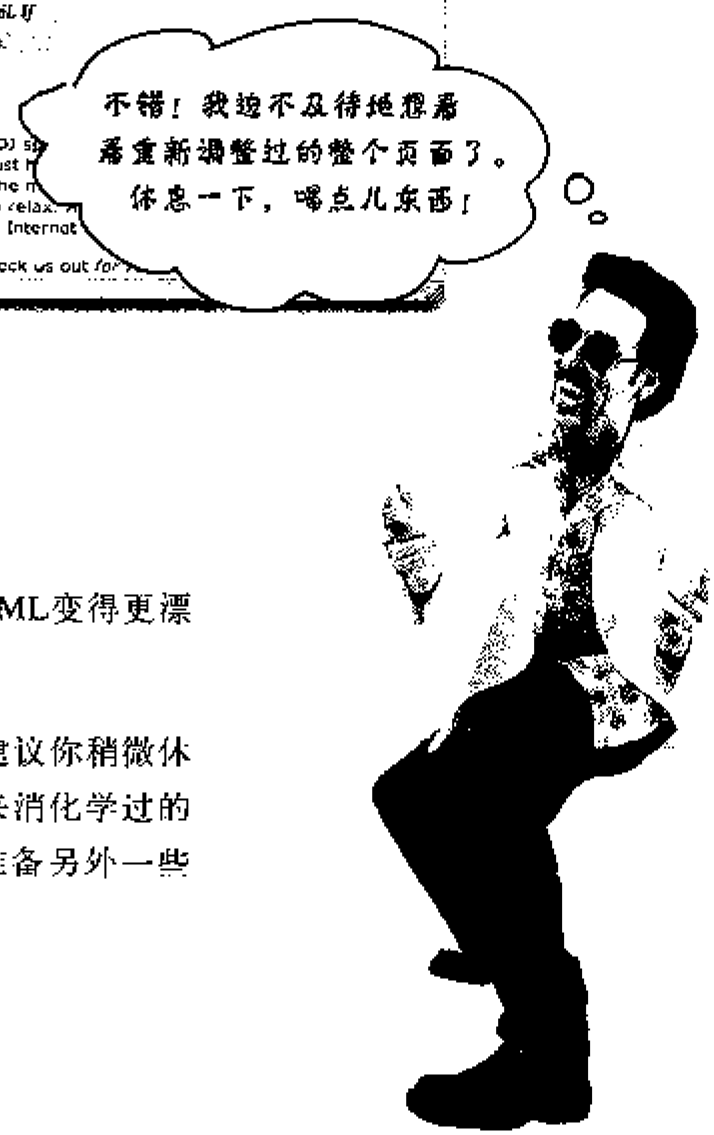


注意！

（浏览器对thin,medium,thick默认的大小不一定总是相同的。）

浏览器可能对关键字thin,medium,thick有不同的默认大小，所以如果边框大小对你确实很重要，考虑用像素大小代替。



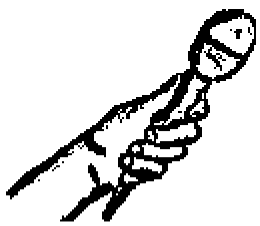


祝贺你!

好啊!妙！你只用15行CSS就把一段普通的HTML变得更漂亮、更生动了！

到达这里是一段很长的旅程，所以现在我们建议你稍微休息一下。给自己倒一杯冰奶茶，花一些时间来消化学过的东西——当你缓过神来的时候，我们会为你准备另外一些CSS的重点。

欢迎回来，来的正是时候。我们正要听一个对XHTML类的采访……



类揭秘

本周访问：
用类总是恰当的吗？

Head First：你好，类，你知道我们一直在很好地使用你，但我们还是不太了解你。

Class：我一点都不复杂。这么说吧，如果你想创建一个可以样式化的“组”，只要创建一个类，把元素都放进去，就可以一起样式化里面的元素了。

Head First：所以用类可以给一系列元素定义一个或多个样式属性？

Class：是的。假如你的页面中有一些以假期为主题的部分，一个是Halloween,一个是Christmas，你可以把所有有关Halloween的元素添加到“Halloween”类，把有关“Christmas”的元素添加到“Christmas”类。然后就可以通过为每个类所写的规则单独地样式化这些元素了，比如Halloween为桔红色，Christmas为红色。

Head First：这似乎很有意义。我们在这章中看到了一个很好的例子，不是吗？

Class：不清楚，我刚好不在场。你得提醒我一下。

Head First：在Head First 休闲室页面中有一段是主人写的保证语，他们想让这段在整个页面中显得突出一些。

Class：听起来不错……不过我问你一个问题：这样的段落有几个还是只有一个？

Head First：我觉得没有必要有几个保证段，并且我没有看到页面中其他地方应用了同样的样式，所以只有一个。

Class：嗯……我不喜欢那样的说法。你知道，类是要应用一种特殊样式到多个元素。如果只有单独的一个元素需要样式化，不是类真正要做的。

Head First：等一下——一个类好像工作得很完美……这怎么会不对呢？

Class：别那么惊讶。你需要做的只是把`class`属性换成`id`属性，只要花一点时间就行。

Head First：**id**属性？我觉得它只用于URL的目标锚中，就像在第4章？

Class：**id**有许多用途。它们其实只是每个元素唯一的标识符。

Head First：你能多告诉我们一些有关**id**属性的内容吗？这些内容对我来说都是从未接触过的。我的意思是，整个一章我都用错了类。

Class：别担心，这是个常犯的错误。一般你只需知道当你想应用一种样式到多个元素时就用类。相反，如果你需要应用样式的只是一个元素并且页面上只有一个，就用**id**。严格来说**id**属性是命名唯一的元素的。

Head First：好的。我想我明白了，但是为什么我们需要**id**呢？我的意思是，类也做得很好啊。

Class：因为有一些东西你其实想在页面上只要一个。你提到的保证段就是个例子；不过还有更好的例子，像页面中的标题、页脚或导航条。你不会希望这些东西在网页中有两个。当然，你可以只给一个元素用类，但会有别人给这个类添加别的元素，那么你的元素就不再有唯一的样式了。如何放置HTML元素也很重要，不过你还没接触到。

Head First：那么，好的，类。这次谈话确实对我们很有教育性。听起来好像我们绝对需要把那段从类变为**id**。谢谢你。希望您还能接受我们的采访。

Class：随时都可以，Head First！

BRAIN POWER

给以下的元素选择要用class还是id：

id class

❑ ❑ 含有页脚的一个段落。

❑ ❑ 一系列含有公司传记的标题和段落。

❑ ❑ 含有图名为“一天美景”的一个**<img>**元素。

id class

❑ ❑ 一系列含有电影回顾的**<p>**元素。

❑ ❑ 一个**<ol>**元素,里面列出了你要做的事。

❑ ❑ 含有Buckaroo Bonzai引用的一些**<q>**元素。

答案：页脚，一天的照片，要做的事列表是适用id的最佳候选人。

id属性

因为你在<a>元素中用过**id**，并且已经知道如何使用**class**属性了，所以你不用学习太多就能使用**id**属性。假如你的页面要有个页脚，通常在一个页面上只有一个页脚，所以选**id**好像很合适。给一个有页脚文本的段落添加标识符“footer”，方法如下所示：

和类一样，只需添加属性“id”并选择一个唯一的id名。

和类不同的是，页面中只能有一个元素的id为“footer”。

```
<p id="footer">Please steal this page, it isn't copyrighted in any way</p>
```

每个元素只能有一个id。

id名称必须以字母打头，后跟数字或字母。不允许有空格或特殊字符。

给元素设置id和把元素添加到类中的操作类似。仅有的不同之处是属性叫做“id”，而不是“class”，一个元素不能有多个id，而且同一页面中不能有一个以上的元素用同一个id。

there are no Dumb Questions

问：有什么大不了的？为什么仅仅为了证明页面中的某些东西是唯一的，就需要一个id呢？我可以同样地使用类，对吗？

答： 可以用类“冒充”id，但阻止你这么做的原因有很多。假如你跟一组人一起做一项网页工程。你的一个队友要是看到一个类就会觉得可以应用到别的元素。反过来，如果她看到一个id，就知道是标识唯一的一个元素的。还有许多别的原因表明id的重要性，后两章你就会看到。例如，当你开始在页面中放置元素时，要放置的每个元素都需要一个唯一的id。

问：一个元素可以既有一个id又属于一个类吗？

答： 可以。可以这么想：id只是一个元素唯一的标识符，但这不意味着它不能属于一个或多个类（就像有一个唯一的名字不意味着你不能加入一个或几个俱乐部）。

在CSS中如何使用id?

选择一个具有id的元素和选择一个具有类的元素一模一样。快速复习一下：如果有一个叫做“specials”的类，用这个类选择元素的方法有很多。你可以在类中选择特定元素，像这样：

```
p.specials {
      color: red;
}
```

这条规则只选择specials类中的段落。

或者，可以选择属于“specials”类的所有元素，像这样：

```
.specials {
      color: red;
}
```

这条规则选择specials类中的所有元素。

用id选择符是十分相似的。要通过一个元素的id选择它，在id前面加一个“#”（跟类比较，在类中类名称之前用一个“.”）就行了。假如你想选择id为“footer”的元素：

```
#footer {
      color: red;
}
```

这条规则选择id为“footer”的任意元素。

或者，可以只选择id为“footer”的<p>元素，像这样：

```
p#footer {
      color: red;
}
```

这条规则选择一个id为“footer”的<p>元素。

类和id之间的唯一不同是id选择符只跟一个页面中的一个元素匹配。

在休闲室中使用id

我们的“保证段落”确实应该有一个id，因为这段在页面中只出现一次。我们开始的时候就应该用这种方法设计，不过改起来也不是很麻烦。

第一步：在“lounge.html”中把class属性改为id：

只要把class属性改为id。

```
<p id="guarantee">
    Our guarantee: at the lounge, we're committed to providing
    you, our guest, with an exceptional experience every time you
    visit. Whether you're just stopping by to check in on email
    over an elixir, or are here for an out-of-the-ordinary dinner,
    you'll find our knowledgeable service staff pay attention to every
    detail. If you're not fully satisfied have a Blueberry Bliss Elixir
    on us.
</p>
```

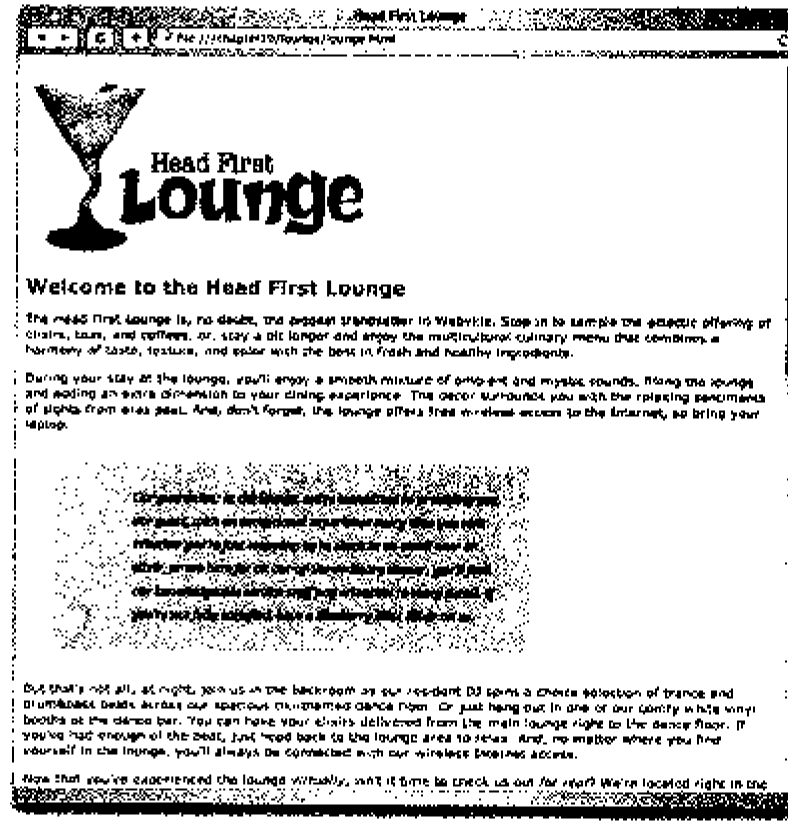
第二步：把“lounge.css”中的“.guarantee”类选择符改为id选择符：

只要把选择符中的“.”改为“#”。

```
#guarantee {
    line-height:           1.9em;
    font-style:            italic;
    font-family:           Georgia, "Times New Roman", Times, serif;
    color:                 #444444;
    border-color:          white;
    border-width:          1px;
    border-style:          dashed;
    background-color:      #a7cece;
    padding:               25px;
    padding-left:          80px;
    margin:                30px;
    margin-right:          250px;
    background-image:      url(images/background.gif);
    background-repeat:     no-repeat;
    background-position:   top left;
}
```

第三步：保存设置并重新加载页面。

好了，一切看起来都完全一样。不过，既然一切都跟我们想要的一样，你不觉得这样更好吗？



there are no Dumb Questions

问： 为什么要用#guarantee而不用p#guarantee?

答： 我们可以用它们中的任何一个，效果都是一样的。因为在这页上我们知道要给id指派一个段落，所以用哪个其实无所谓（#guarantee更简单一些）。然而在一些更加复杂的网页中，在一些页面上，一个id会被指派给一个段落，而在别的页上被指派给列表或块引用。所以这个id需要有几条规则，使用p#someid还是blockquote#someid，取决于页面上是哪种元素。

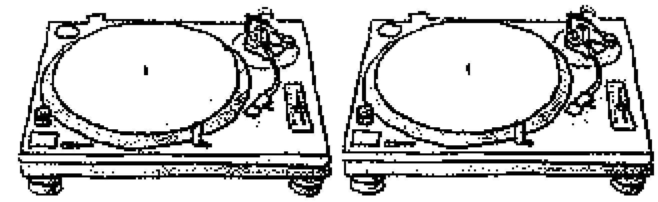
问： 我能不能总是先用类，然后在知道它是唯一的之后再改成id呢?

答： 不能。在你设计页面的时候应该知道一个元素是不是唯一的。我们之所以只在这一章这么做是因为开始的时候你还不知道有id。不过我们这样把id穿插进来介绍，你不觉得是个不错的做法吗？

问： 在第4章，我们在元素<a>中用id属性来创建目标锚。当我把id放进别的元素，它们也成了目标吗?

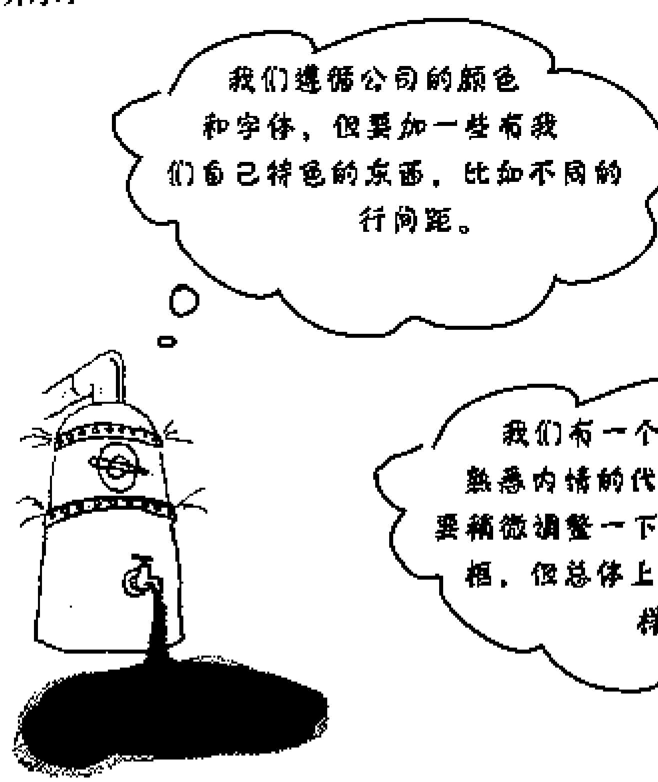
答： 是这样的，现代的大多数浏览器都支持，但以前的浏览器不支持。

重新混合样式表

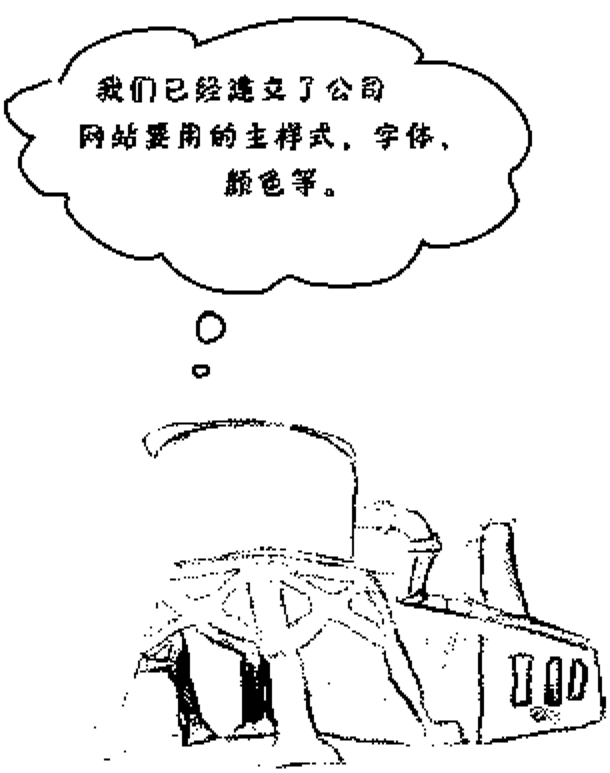


在我们结束这一章之前，给一些样式表来一个有趣的组合。到现在为止你一直只用一个样式表。谁说过你只能用一个样式表呢？你可以定义一系列的样式表用于任何XHTML。但是或许你会想为什么有人想用多个样式表呢？有许多充分的理由，以下是第一个……

假如Head First休闲室开张了，有了公司权，做了首次公开募股等（当然都是由于你和你的一流网站的功劳）。那么就会有一个包含上百个页面的完整的公司网站，很显然你想用外部CSS样式表样式化这些页面。会有许多不同的公司部门，他们想用各自不同的方法调整这些样式。休闲室的经营者们也想控制样式，大致如下所示：



饮料部门



公司



西雅图休闲室（饮料部门的一个分部）

使用多样式表

那么你如何先设计好一个公司的样式，然后允许各部门和休闲室经营者们去任意改变样式呢？这就得用几个样式表，像这样：

在XHTML中你可以定义一个以上的样式表，我们定义了三个。

一个样式表是为整个公司设计的。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en" >
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
    <title>Head First Lounge</title>
       <link type="text/css" href="corporate.css" rel="stylesheet" />
       <link type="text/css" href="beverage-division.css" rel="stylesheet" />
       <link type="text/css" href="lounge-seattle.css" rel="stylesheet"  />
  </head>
  <body>
  .
  .
  .
  </body>
</html>
```

饮料部门可以给公司样式添加一些内容，甚至可以覆盖一些公司样式。

西雅图休闲室也可以在他们的样式表中做自己的调整。

顺序很重要！一个样式表会覆盖在它之前链接的样式表中的样式。

there are no Dumb Questions

问： **样式表的顺序很重要，是吗？**

答： 是的，它们从上到下排列，最下面的样式表优先权最高。所以假如在公司和部门的样式表中，<body>元素都有一个font-fanmily属性，则部门的样式表就有较高的优先权，因为它是最后一个链接到XHTML的。

问： **在简单的站点中我需要多样式表吗？**

答： 用它会给你带来惊喜。有时你看到一个样式表，想把它作为页面的基础，你不用改变样式表，只要链接它，再把你自己的样式表放到下面来定义你想改变的内容就行了。

问： **可以再讲一下如何确定一个具体元素的样式吗？**

答： 我们在第8章中讲过一点。现在，要知道的是样式表链接文件的顺序很重要。然后在下一章中，在学了另一些CSS的细节后，我们将深入学习浏览器如何知道该给哪个元素应用哪种样式。

样式表——它们不再仅仅用于桌面浏览器……



现在是你要有多个样式文件的第二个理由——让电脑屏幕、PDA或移动设备以稍有不同的格式输出你网页的不同版本。**<link>**元素有一个叫做**media**的属性，可以用来定义你的样式文件所需要的设备类型。

如果你想知道更多关于输出格式和移动设备类型的信息，查看附录的有关部分。

Media属性可以用来定义应用这个样式表的设备类型。

```
<link type="text/css" rel="stylesheet" href="lounge-screen.css" media="screen" />
```

这儿我们定义这个样式表适用于电脑屏幕。

在同一个XHTML文件中用不同的媒体类型定义多个**<link>**元素，方法如下：

如果没有提供媒体种类，样式文件就应该适用于所有设备。

```
<link type="text/css" href="lounge.css" rel="stylesheet" />
<link type="text/css" href="lounge-print.css" rel="stylesheet" media="print" />
<link type="text/css" href="lounge-mobile.css" rel="stylesheet" media="handheld" />
```

现在有了另外两个<link>元素，一个定义输出，一个定义具有小屏幕和有限信息速度的小设备。

there are no Dumb Questions

问： 真酷！我可以为不同的设备创建不同的样式表，是吗？

答： 是的，你可以创建几个样式表，然后把它们都链接到你的XHTML。浏览器就会根据媒体类型选择合适的样式表。

问： 假如有一个链接"handheld"和一个应用到所有浏览器的链接，会应用哪个呢？

答： handheld浏览器会两个都应用。不过，假设"handheld"链接在"所有"链接后面，handheld规则就会有优先权，就像我们以前在公司、部门和lounge CSS文件中提过的一样。

问： 除了屏幕（电脑）、打印（类似打印机的媒体）、handheld（移动设备和移动电话），还有什么媒体类型呢？

答： 还有另外一些：aural（应用于高速浏览器），Braille（给需要读盲文的人），projection，tty（电传打字机和终端设备），tv（电视）。

练习

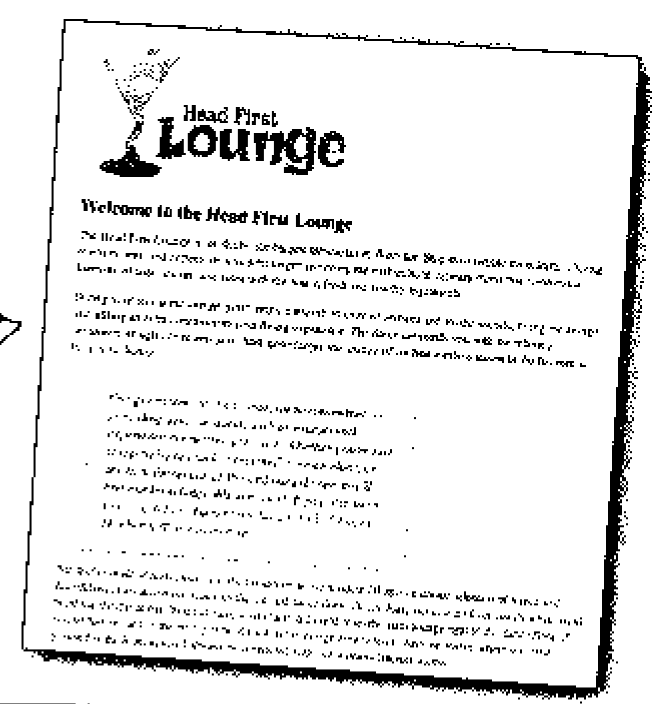
在"chapter10/lounge/print"目录下，你会发现"print.css"。打开"chapter10/lounge"目录下的"lounge.html"，给这个样式表添加新的链接，媒体类型为"print"。同时在链接了"lounge.css"的<link>元素中添加属性media="screen"，那么你就有了两个样式表，一个给屏幕，一个给打印机。然后保存，重新载入页面，选择浏览器的"打印"选项。运行打印机看看结果。

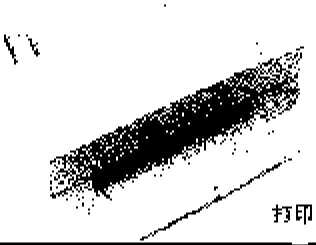
```
<link type="text/css" href="print/print.css"
      rel="stylesheet" media="print" />
```

这是你需要添加到"lounge.html"文件的新链接。

这是打印的版本。你用CSS完全改变了页面打印后的外观。这种结构及外观都很成功。

不幸的是，并不是所有的浏览器都支持media属性，所以如果你得不到这个结果，换一个浏览器试试。





打印机是可选的设备，不一定非要有。

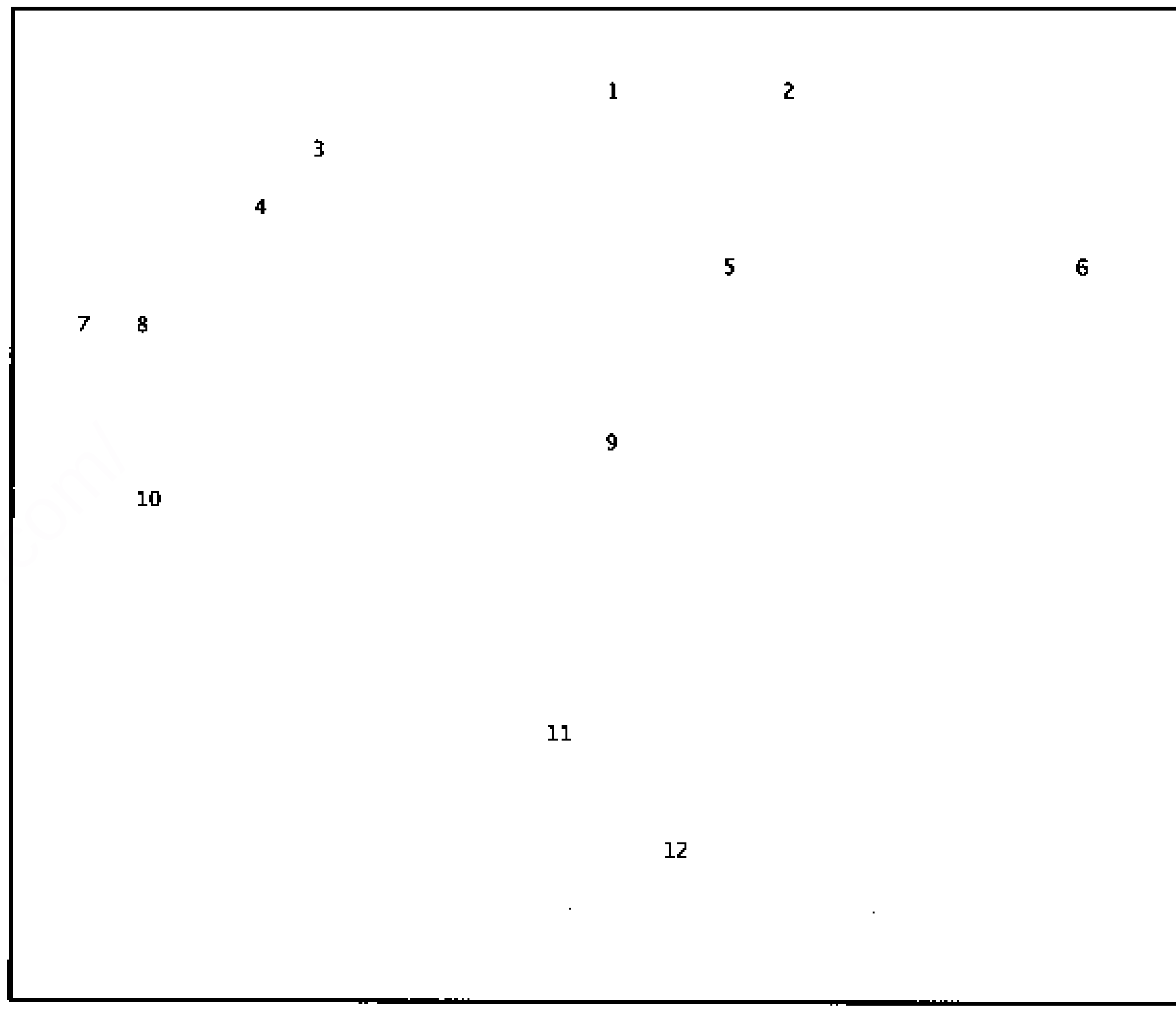
要点

- CSS用盒模式控制元素显示的方式。
- 盒子由内容区和可选的补白、边框和边界组成。
- 内容区是元素的内容。
- 补白用来在内容区周围创建可见的空间。
- 边框包围着补白和内容，是可见的，将内容分隔开。
- 边界包围着边框、补白和内容，用来在元素和元素之间添加空间。
- 补白、边框和边界都是可选的。
- 元素的背景可以在内容和补白底下显示，但不能延伸到边界。
- 可以用像素或百分数设置补白和边界的尺寸。
- 可以用像素或关键字thin、medium和thick设置边框宽度。
- 边框有8种不同的样式，包括solid、dashed、dotted和ridge。
- 设置边界、补白或边框时，CSS提供同时设置各个侧面的属性（上，下，左，右），也可以单独设置。
- line-height属性用来增加文本的行间距。
- background-image属性用来给元素添加背景图像。
- background-position和background-repeat属性用来设置背景图像的位置和重复方式。
- class属性用来同时样式化一组元素。
- id属性用来给元素设置唯一的名字。也可以用id属性给一个元素提供唯一的样式。
- 一个页面中一个元素只能有一个id。
- 可以用id#selector选择元素，比如#myfavoriteid。
- 一个元素只能有一个id，但可以属于几个类。
- 可以在XHTML中用多个样式表。
- 如果两个样式表有冲突的属性定义，XHTML文件中靠后的样式表具有优先权。
- 可以在<link>元素中用media属性确定媒体设备类型，如“print”或“handheld”。



XHTML填字游戏

你确实扩展了你的HTML和CSS技能。现在通过做填字游戏来加深你对知识的理解与巩固。答案都来自本章。



横排提示：

2. CSS把每一个元素看作一个______。

4. 在保证段中首选的字体样式（font-style）。

5. 用于______其他类型的可选的<link>属性。

7. 位于补白和边界之间。

9. 我们把______类改为id。

10. 如果想让元素有一个唯一的样式，用这种选择符。

11. 内容和边框之间的空间。

12. 在保证段落中用的边框的样式。

竖排提示：

1. 行之间的空间的出版术语______。

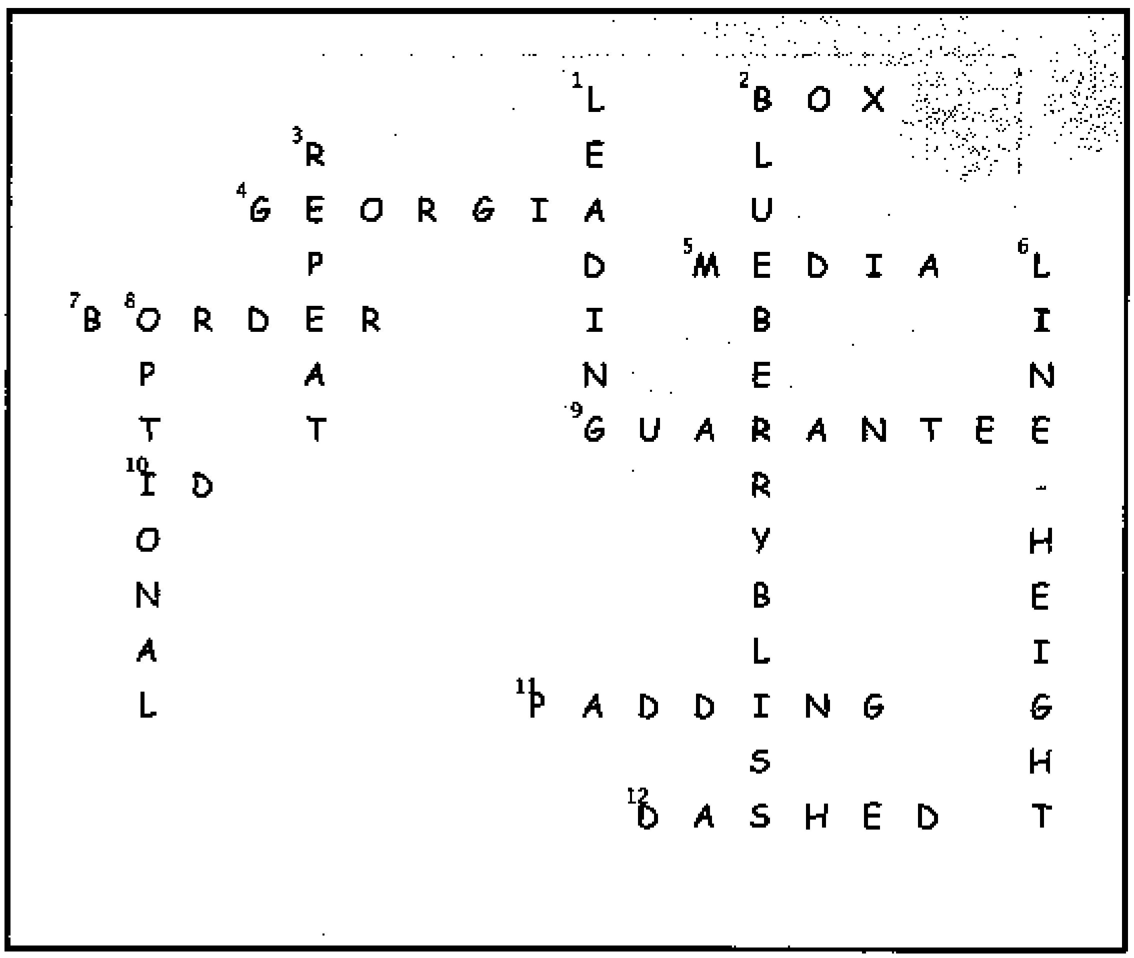
2. 如果你不完全满意的话选择哪种饮料。

3. 默认的背景图像是这样的。

6. 用来增加文本行之间的空间的属性。

8. 补白、边框和边界都是______。

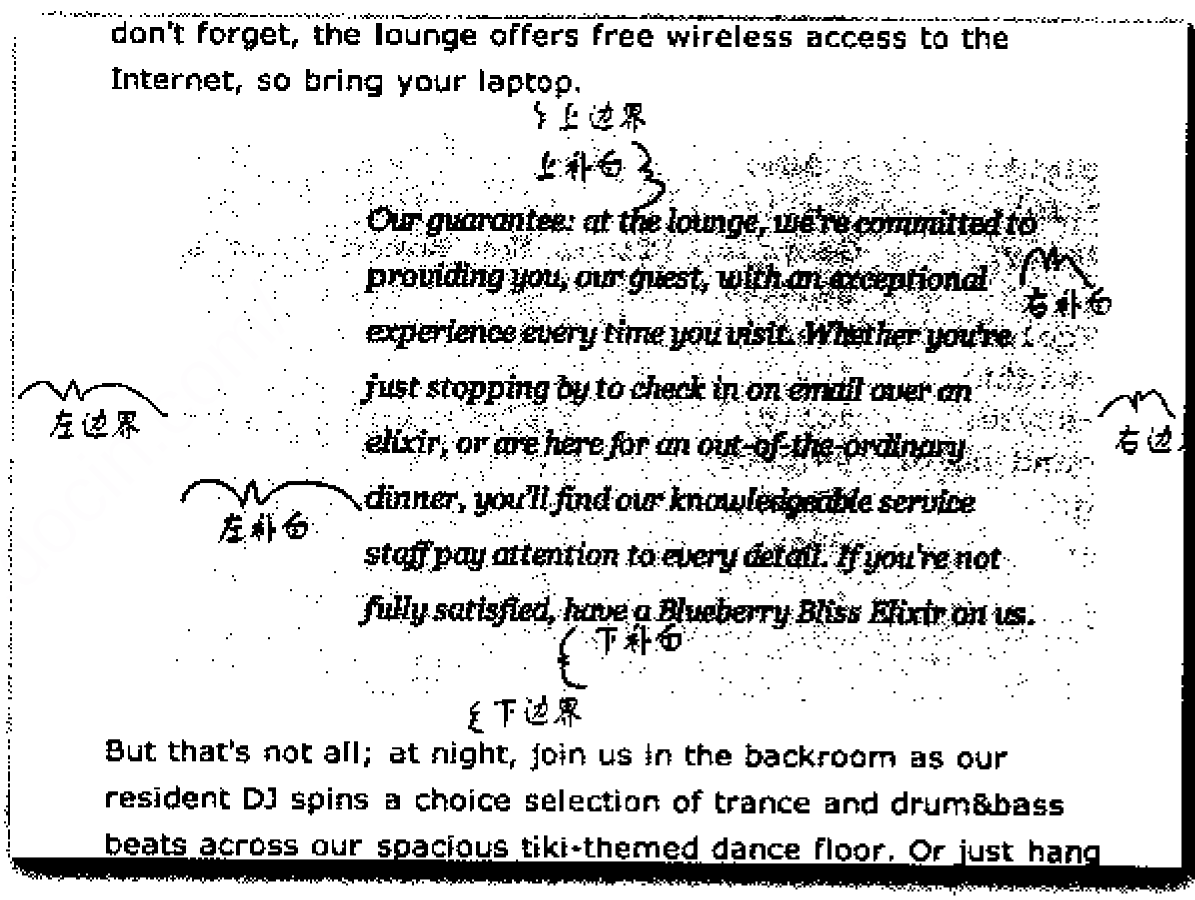
XHTML填字游戏解答



Sharpen your pencil

答案

看看你能否区别这段的补白、边框和边界。标记出所有的补白和边界（上，下，左，右）：



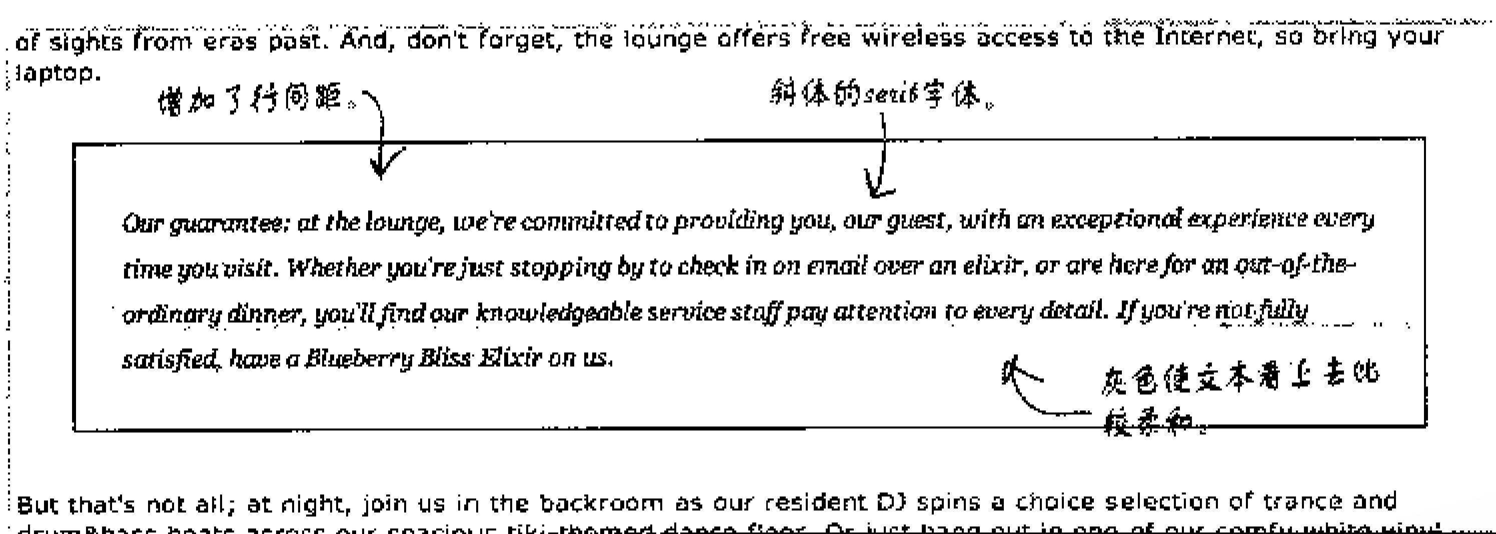
练习答案

看一下最后一种版本的保证段落，它的字体是斜体、serif，行间距比页面中其他文本的要大。如果再看仔细一些，文本是灰色的。按以下要求写CSS，把行间距设置成1.9em，字体样式为italic，颜色为#444444，字体列表为Georgia、“Times New Roman”、Times、serif。以下是答案，你测试了吗？

可以把新属性加在规则的任意位置，我们加在最前面。

```
.guarantee {
        line-height:        1.9em;
        font-style:         italic;
        font-family:        Georgia, "Times New Roman", Times, serif;
        color:              #444444;
        border-color:       black;
        border-width:       1px;
        border-style:       solid;
        background-color:   #a7cece;
        padding:            25px;
        margin:             30px;
}
```

注意如果字体名字中间有空格，应该加引号。



11 div和span

高级网站构建



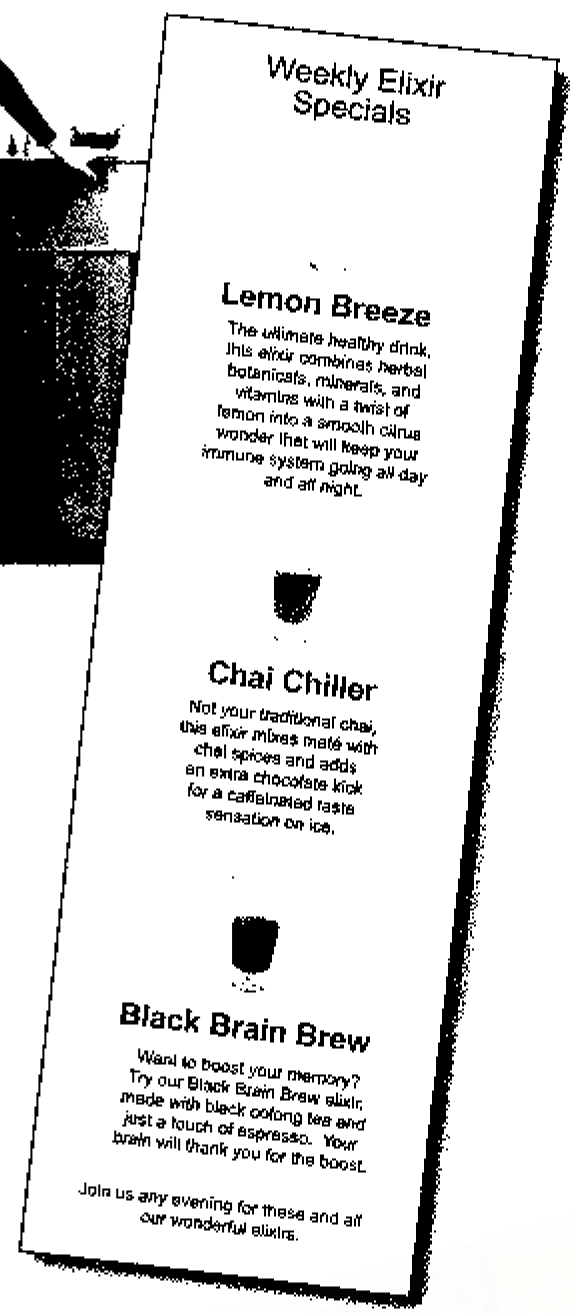
该来点高级建筑了。 这一章我们将引出两个新的XHTML元素，叫做<div>和<span>。这些不是“微不足道”的小东西，而是到处都要用到的钢铁横梁。你将用<div>和<span>构建一些严格的支持结构，一旦把这些结构放到合适的位置，就能用全新而强大的方式样式化它们了。现在，我们还不能帮你这个忙，因为我们发现你的CSS工具箱已经开始填满了，所以首先要告诉你一些捷径，以便更加轻松地指定这些属性。此外，这一章有一些特别的客人——伪类，你可以用它们创建一些非常有趣的选择符（如果你觉得伪类”可以作为你下一个乐队的名字，那么太晚了，我们抢先了一步）。

你知道，如果能把网页上的特制饮料做得更诱人一些我们会更喜欢它。你能把它做得跟我们散发的传单一样吗？

调酒师，Alice



这是有特制饮料的散发传单。设计得跟页面的其他部分很不一样：很窄，文本居中，标题是红色的，浅绿色的边框包围着所有的内容，顶部甚至还有一些鸡尾酒杯的图形。



Weekly Elixir Specials

Lemon Breeze

The ultimate healthy drink, this elixir combines herbal botanicals, minerals, and vitamins with a twist of lemon into a smooth citrus wonder that will keep your immune system going all day and all night.

Chai Chiller

Not your traditional chai, this elixir mixes maté with chai spices and adds an extra chocolate kick for a caffeinated taste sensation on ice.

Black Brain Brew

Want to boost your memory? Try our Black Brain Brew elixir, made with black oolong tea and just a touch of espresso. Your brain will thank you for the boost.

Join us any evening for these and all our wonderful elixirs.

仔细观察饮料XHTML

Alice的确提出了很高的要求，不是吗？她想让我们把现有的休闲室XHTML做成散发的传单样子。这看起来很有挑战性，不过正因为我们有CSS，所以就让我们来试一试吧。但是，在投入样式化工作之前，我们先大体看一下现有的XHTML。以下是特制饮料的XHTML摘录，你会在“chapter11/lounge”目录下的“lounge.html”文件中找到：

```
<h2>Weekly Elixir Specials</h2>

<p>
    <img src="images/yellow.gif" alt="Lemon Breeze Elixir" />
</p>
<h3>Lemon Breeze</h3>
<p>
    The ultimate healthy drink, this elixir combines
    herbal botanicals, minerals, and vitamins with
    a twist of lemon into a smooth citrus wonder
    that will keep your immune system going all
    day and all night.
</p>

<p>
    <img src="images/chai.gif" alt="Chai Chiller Elixir" />
</p>
<h3>Chai Chiller</h3>
<p>
    Not your traditional chai, this elixir mixes mat&eacute;
    with chai spices and adds an extra chocolate kick for
    a caffeinated taste sensation on ice.
</p>

<p>
    <img src="images/black.gif" alt="Black Brain Brew Elixir" />
</p>
<h3>Black Brain Brew</h3>
<p>
    Want to boost your memory? Try our Black Brain Brew
    elixir, made with black oolong tea and just a touch
    of espresso. Your brain will thank you for the boost.
</p>

<p>
    Join us any evening for these and all our
    other wonderful
    <a href="beverages/elixir.html"
       title="Head First Lounge Elixirs">elixirs</a>.
</p>
```

特制饮料部分从标题<h2>开始。

有3种饮料，它们结构相同。

每一种饮料在<p>元素中有一幅图像。

……在标题<h3>中有一个名字……

……

在段落中有一段描述。

每种饮料都重复这种结构。

最后，末尾还有一段，是一些文本和一个到真正饮料页面的链接。



Jim：别担心，Frank，我们可以创建一两个类，然后单独给所有的饮料元素添加样式。

Frank：可以这么做，也许还不是那么糟糕。我知道有一个使文本居中的简单属性，并且我们知道如何处理文本的颜色。

Jim：等一下，包围着所有内容的边框怎么办？

Frank：小意思，我们刚学过怎样做边框。别忘了，每个元素都能有一个。

Joe：我不那样认为。如果你看一下XHTML代码就会发现这些是**<h2>**，**<h3>**和**<p>**元素。如果我们给每一个元素都单独加边框，它们看起来就会像分开的盒子。

Frank：你说得对，Joe。我们需要的是一个能把所有这些元素都嵌入进去的元素，所以我们可以弄一个边框，把页面上的所有饮料部分包围起来。

Jim：太棒了，Frank。我们能不能把这些饮料放进一个**<p>**元素或**<blockquote>**中呢？

Frank：不能用**<p>**，因为**<p>**不能包含别的块元素，标题和段落显然是块元素。无论如何这也不是我们所希望的，段落是相对文本而言的。

Joe：**<blockquote>**也不合适，因为这是个饮料传单，不是引用。

Frank：其实我觉得我们的方向是正确的。我正在读一本专门讲解HTML和CSS的书，现在碰到一个叫做**<div>**的新元素，我认为它就是我们需要的工具。

Joe：**<div>**——那是什么？听起来像是数学方面的。

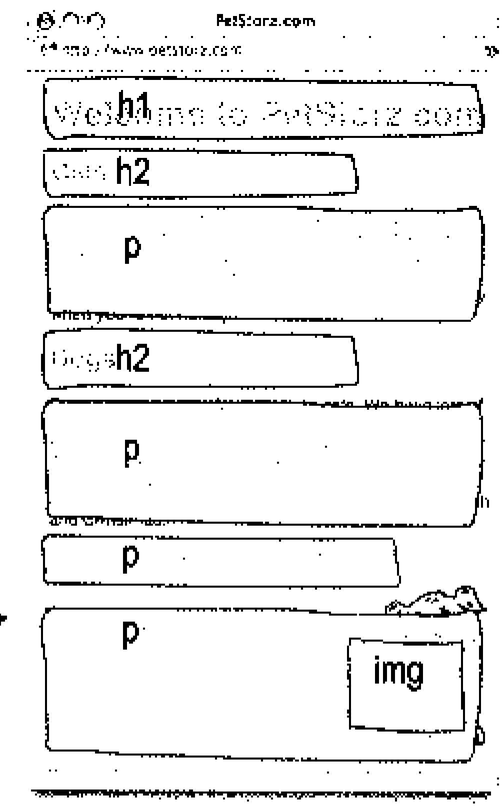
Frank：没有脱离主题，因为**<div>**用来把页面分割成逻辑部分(logical sections)或者组。

Jim：哈!这好像正是我们需要的!

Frank：是的。我来教你们怎么把页面分割成几个逻辑部分，接下来我还会告诉你们我所知道的关于**<div>**的知识……

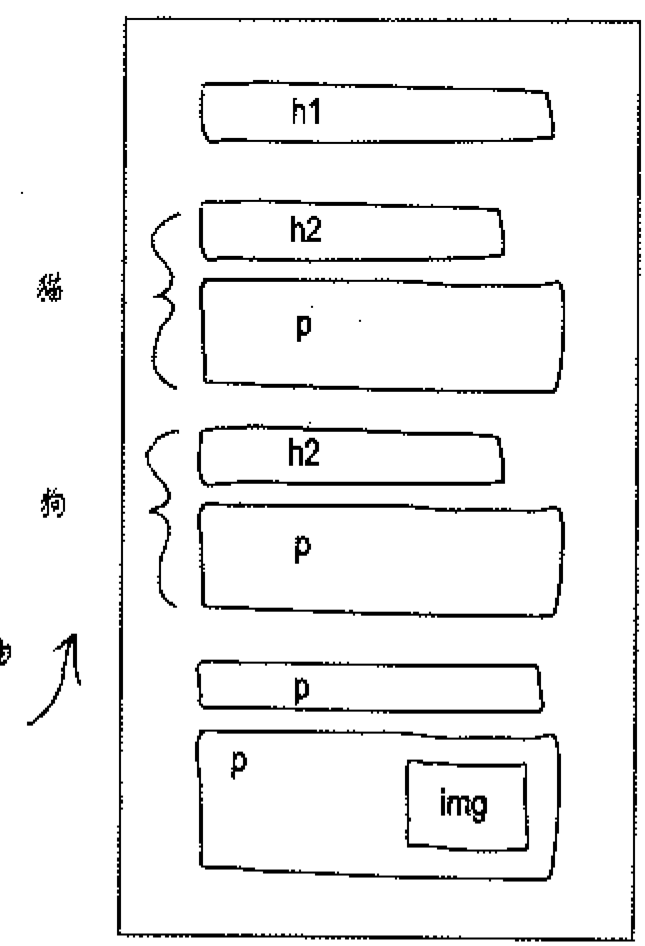
研究一下如何把一个页面分割成几个逻辑部分

看一下右边的网页：这是PetStorz.com中的一个网页，我们将用几页的篇幅介绍如何通过确定一些逻辑部分，然后把它们封装进一个<div>元素来添加另外一些结构。



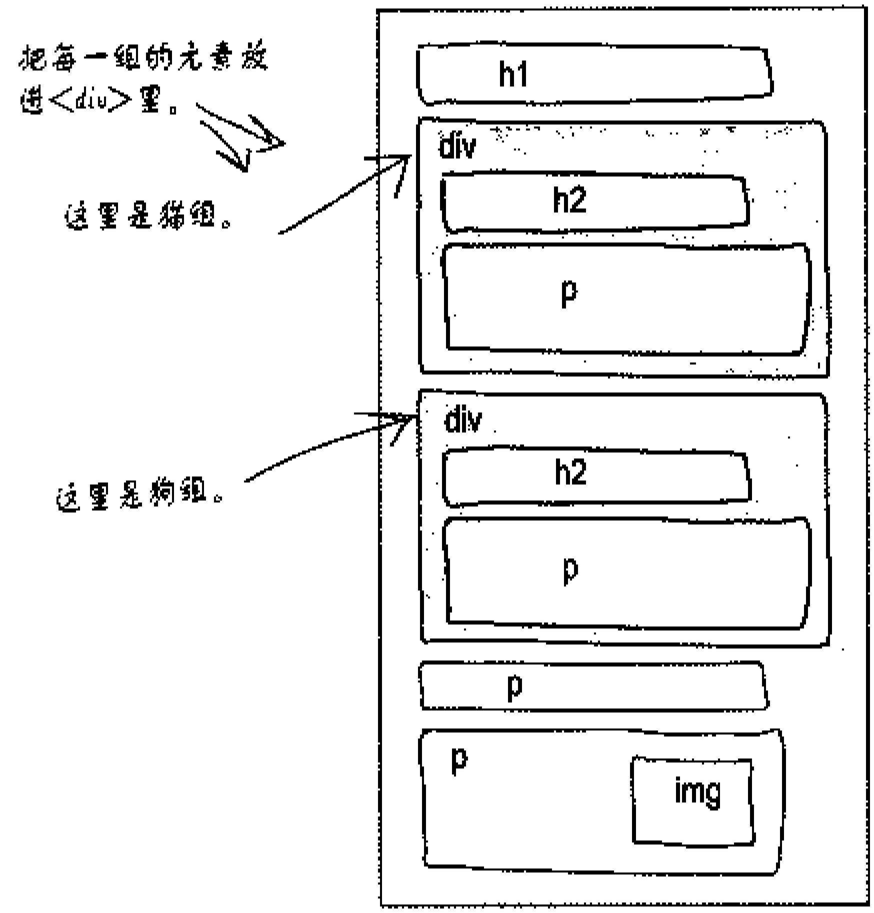
确定逻辑部分

好的，那么你的工作就是在这个页面中定位“逻辑部分（logical sections）”。什么是逻辑部分？它只不过是页面上相互关联的一组元素。例如，在PetStorz.com网页中，页面上有一些用于猫区的元素，还有一些用于狗区的。让我们仔细观察一下。



用<div>标记各部分

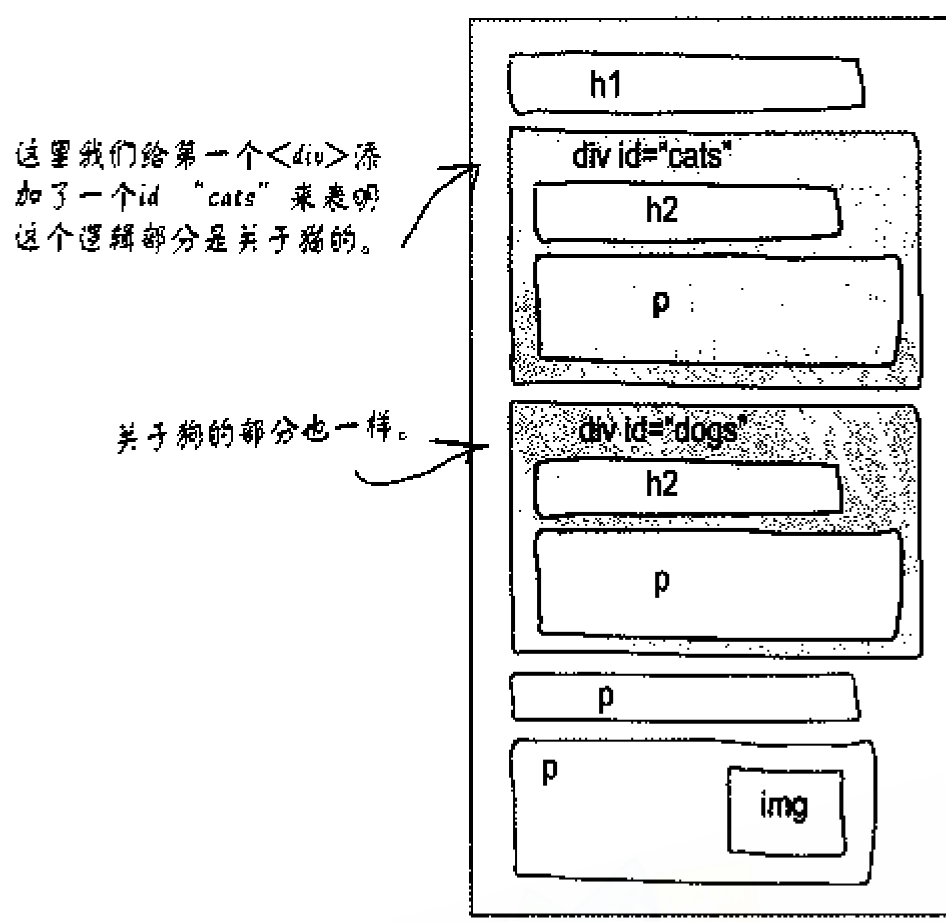
既然你知道哪个元素属于哪一部分了，那么你就可以添加些XHTML来标记这些结构。一般的做法是用**<div>**开始和结束标记把属于一个逻辑部分的元素包围起来。我们先简单地画一下，然后在后面再做真正的标记。



标记<div>

把一些元素放进**<div>**里，就表明了它们都属于同一组。但你还没给它们任何标签来表明这个组是什么含义，对吗？

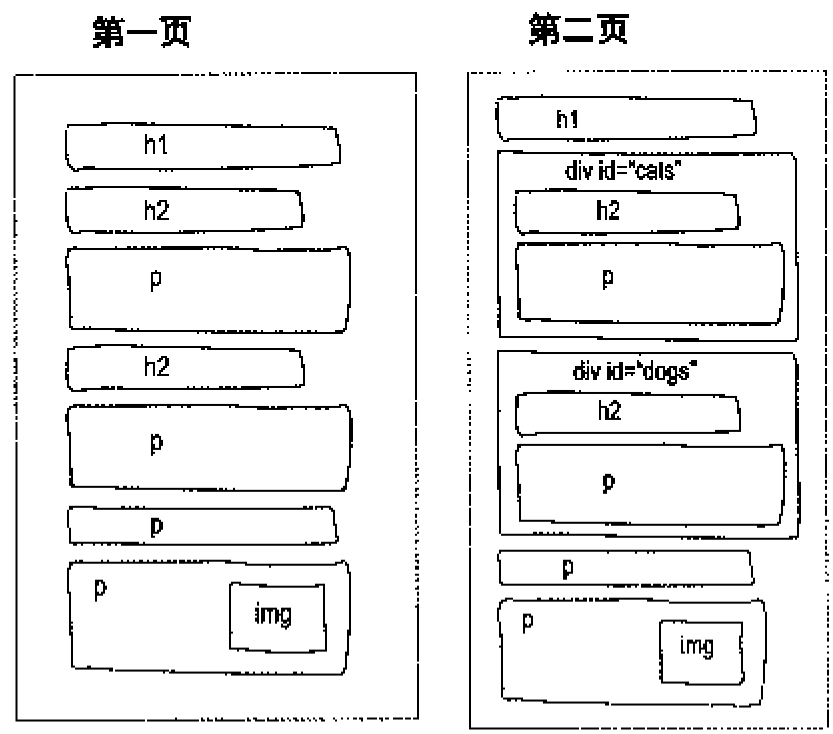
这儿有一种不错的方法来表明一个组的意义——用**id**属性为**<div>**提供一个唯一的标签。例如，给猫**<div>**一个id "cats"，给狗**<div>**一个id "dogs"。



BRAIN POWER

Starbuzz CEO把你叫进来咨询PetStorz主页的样式变化。如果给你的是第一页，你花多长时间能理解PetStorz网页？

如果给的是第二页呢？



添加一些样式

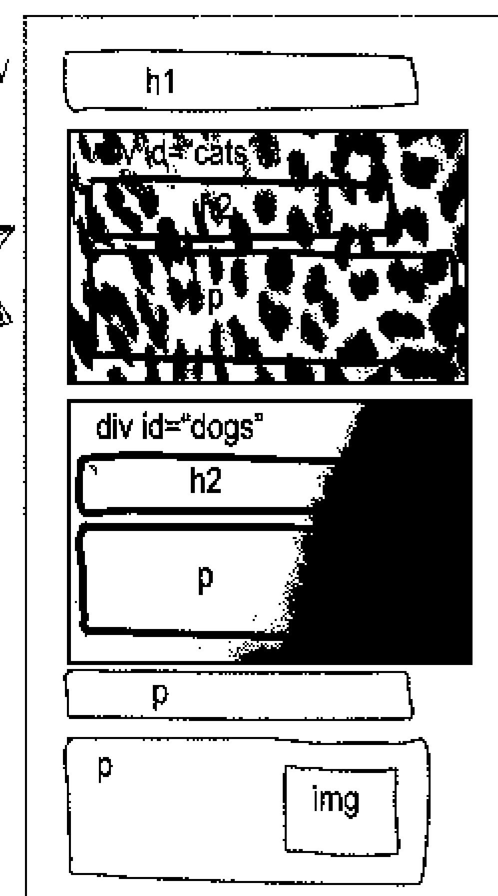
好，你已经给PetStorz页面添加了一些逻辑结构，而且已经通过给每个**<div>**一个唯一的id标记了结构。就先给包含在**<div>**中的一组元素设计一些样式吧。

现在<div>有了一些样式。

设置<div>的背景，使它包含在<div>中的元素也显示背景。

跟其他任何子元素一样（像font-size，color等），<div>中的元素也能从<div>继承一些属性。

这里有两条规则，每个<div>一条。每个id选择符选择一个<div>。




```
#cats {
   background-image: url(leopard.jpg);
}

#dogs {
   background-image: url(mutt.jpg);
}
```

每条规则都设置了background-image属性。给猫设置一个leopard图像，给狗设置一个mutt图像。

展示更多结构

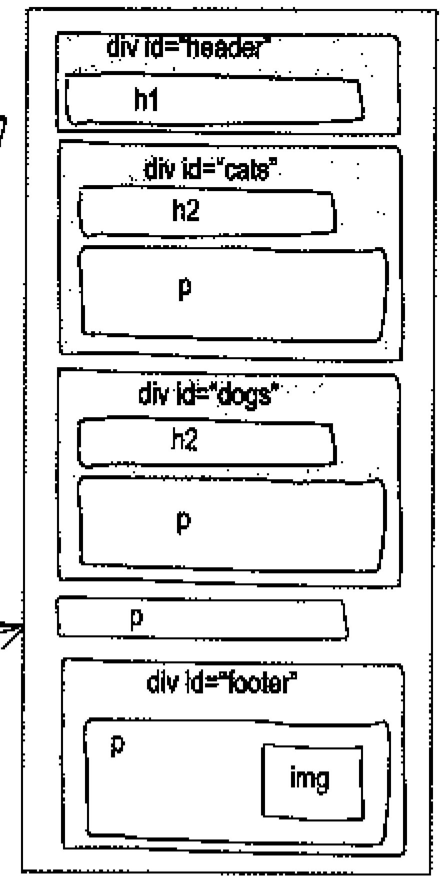
用<div>给页面添加更多结构的原因有很多。首先，你可能想更深一步地展示一下你页面的基本逻辑结构，这样有助于别人理解网页，也有助于维护网页；第二，你可能经常需要用结构把样式应用到某一部分。通常，这两个原因都有。

在PetStorz这个例子中，我们可以提高到另一层次，添加更多的<div>……

现在我们又添加了一个有id的<div>来表明这是页面的标题。

另一个用来表明页面的页脚。

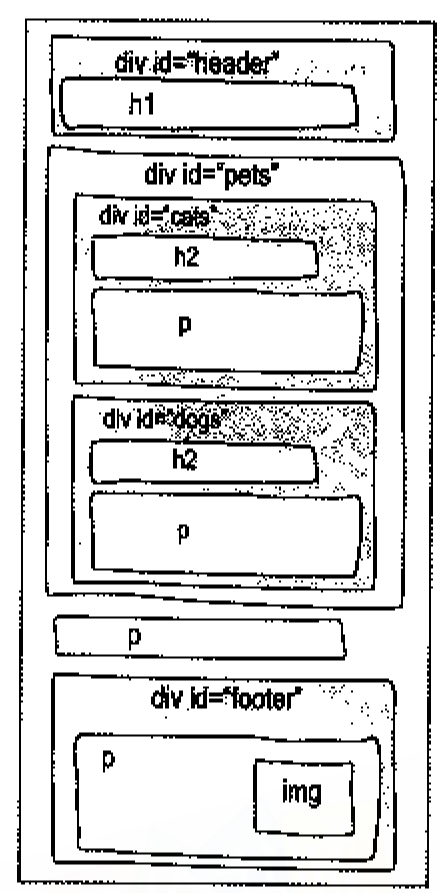
用<div>添加这个结构有助于你思考页面设计。例如，这儿真的需要这个孤立的<p>吗？



在结构上添加结构

还没结束呢，嵌套结构也是很常见的。例如，在PetStorz页中，有一个猫部分和一个狗部分，两部分一起就是页面中逻辑上的“pets”部分。所以，我们可以把“cats”和“dogs”<div>都放进“pets”<div>中。

现在我们标记了这个XHTML，就知道页面中有一个逻辑部分，里面有“pets”内容。更深一层，这个“pets”部分有两个逻辑子部分，一个是“cats”，一个是“dogs”。



there are no Dumb Questions

问： <div>就像一个容器，可以把元素一起放进去，对吗？

答： 的确是这样。其实我们通常把<div>描述为“containers（容器）”。它们不仅可以作为包含一系列相关元素（像“cat”元素）的逻辑容器，而且当我们开始样式化<div>并在下一章中用它们排版时，你会发现它们还可以作为图形容器。

问： 除了结构，我还把标题和段落等放进了页面，我应该用<div>添加一个更高一级的结构吗？

答： 是，但又不是。你可以在真正需要的时候添加结构，而不是为了结构而添加结构。只要能完成任务，结构越简单越好。例如，给PetStorz页添加一个包括“cats”和“dogs”的“pets”部分很有用，那就尽管添加吧，然而，如果没有真正的好处，就只会把页面弄得更复杂。用<div>工作一段时间后，你就会掌握什么时候使用及使用多少<div>了。

问： 可以把<div>放在一个类里而不给它设置id吗？

答： 当然，别忘了一个元素可以有一个id，可以同时在一个或多个类中，所以这种选择并不互相排斥。并且，创建<div>并放进类里的情形很多。假如在一个音乐播放列表页面中有一系列音乐专辑，你会把组成一个音乐专辑的所有元素放进一个<div>中再把它们放进一个叫做“album”的类中。这样就能指明专辑在哪里，还可以通过类给它们设计样式。同时你可以给每个专辑一个id，以便可以单独应用另外的样式。

问： 我在<div>中使用<div>时，如在“cats”、“dogs”和“pets”的使用方面有些困难。能多讲一些这方面的内容吗？

答： 当然。你已经习惯于元素的嵌套，对吗？比如<p>嵌套在<body>中，它们又嵌套在<html>元素中。你甚至看到过列表嵌套列表。<div>也一样，只是把一种元素嵌套进另一种元素，并且，在例子PetStorz中，我们用它来表示更大块的结构（一个“cats”和一个“dogs”嵌套在一个“pets”部分中）。或者，你可以用<div>把一个啤酒部分嵌套进一个饮料部分，再嵌套进一个菜单部分。

不过，要理解<div>的嵌套，最好的方法是进行必要的实践。先把这些抛到脑后吧，你很快就会看到一个这样的例子。

在你的页面中使用<div>，但不要滥用。借助结构，把页面分成几个合理的逻辑结构，这样有助于网页结构的清晰和样式化。如果添加<div>只是想使页面中有更多的结构，那么这样做除了使页面变得复杂之外没有任何真正的好处。

回到休闲室……

好了，关于<div>有足够的“理论”了——我们用休闲室页面做些实践吧。别忘了，我们要把所有的饮料元素组成一组，然后给它设计传单那样的样式。

现在，打开“chapterll/lounge”文件夹下的“lounge.html”文件，找到饮料元素，然后在它们周围添加开始和结束<div>标记。

```
<div id="elixirs">
    <h2>Weekly Elixir Specials</h2>

    <p>
        <img src="images/yellow.gif" alt="Lemon Breeze Elixir" />
    </p>
    <h3>Lemon Breeze</h3>
    <p>
        The ultimate healthy drink, this elixir combines
        herbal botanicals, minerals, and vitamins with
        a twist of lemon into a smooth citrus wonder
        that will keep your immune system going all
        day and all night.
    </p>

    <p>
        <img src="images/chai.gif" alt="Chai Chiller Elixir" />
    </p>

    <h3>Chai Chiller</h3>
    <p>
        Not your traditional chai, this elixir mixes mat&eacute;
        with chai spices and adds an extra chocolate kick for
        a caffeinated taste sensation on ice.
    </p>

    <p>
        <img src="images/black.gif" alt="Black Brain Brew Elixir" />
    </p>

    <h3>Black Brain Brew</h3>
    <p>
        Want to boost your memory? Try our Black Brain Brew
        elixir, made with black oolong tea and just a touch
        of espresso. Your brain will thank you for the boost.
    </p>

    <p>
        Join us any evening for these and all our
        other wonderful
        <a href="beverages/elixir.html"
           title="Head First Lounge Elixirs">elixirs</a>.
    </p>
</div>
```

这是开始标记，我们给它一个叫“elixir”（饮料）的id来识别它。

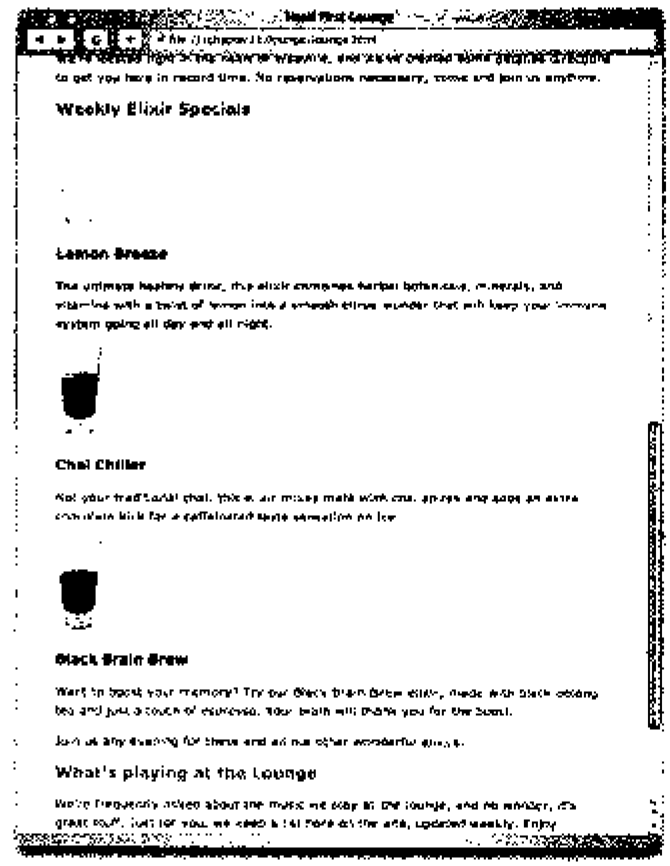
记住我们只是从整个文件中显示了一段XHTML。当打开“lounge.html”时，你会看到页面的所有标记。

这是结束标记。

测试<div>

这很简单，不是吗？既然我们有了更加结构化的页面，就载入浏览器来看看它的效果吧……

根本没变化！不过没关系，<div>仅仅是结构，它在页面中没有任何“外观”或默认样式。

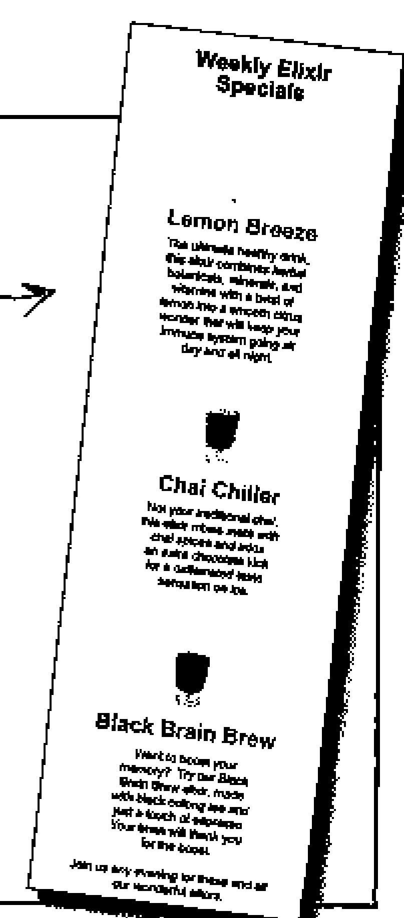


<div>只是个块元素，可以将你需要的任何样式应用于它。所以，只要知道如何样式化一个块元素（你已经知道），就知道如何样式化<div>了。

BRAIN POWER

别忘了，现在的目标是重新样式化页面中的饮料内容，以使它的外观跟传单一样。

我们在绕道去学习<div>之前，就介绍过如何在整个饮料周围得到边框。“lounge.html”中已经有了<div>，就由你来添加边框吧。



添加边框

好了，既然已经有`<div>`包围着饮料部分所有的元素，有趣的事就开始了：可以给它设计样式。

我们想重新设计饮料传单，首先要设置一个包围着饮料部分所有元素的边框，对吗？事实上你已经有了包围着饮料部分的`<div>`，给它设计样式，添加一个边框就可以。现在我们来试一下。

在休闲室的CSS中添加一条新规则，用`<div>`元素的id选择它。打开“chapter11/lounge”目录下的“lounge.css”文件，把这条规则加到末尾：

```
#elixirs {
        border-width: thin;
        border-style: solid;
        border-color: #007e7e;
}
```

把这条规则加到CSS文件末尾。它用饮料的id选择了饮料<div>元素，添加了一个窄的浅绿色实心边框。

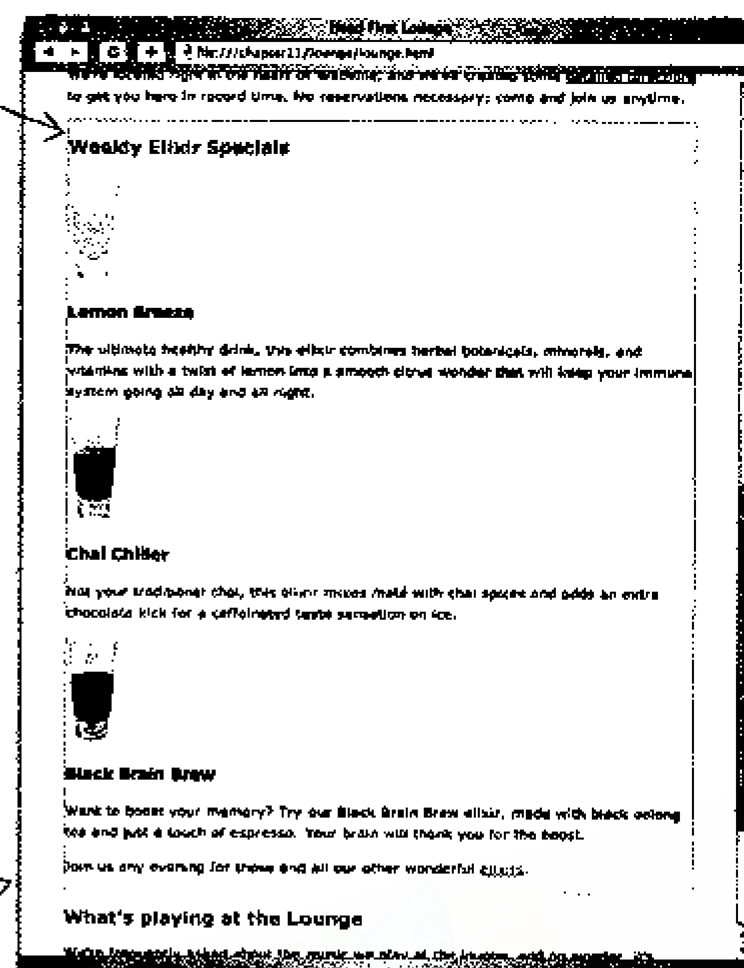
测试边框

改完CSS后，保存并重新载入“lounge.html”文件。

这是你刚才添加到饮料<div>元素的边框。

你给这个<div>添加了可见的边框，不过还必须添加补白和边界。

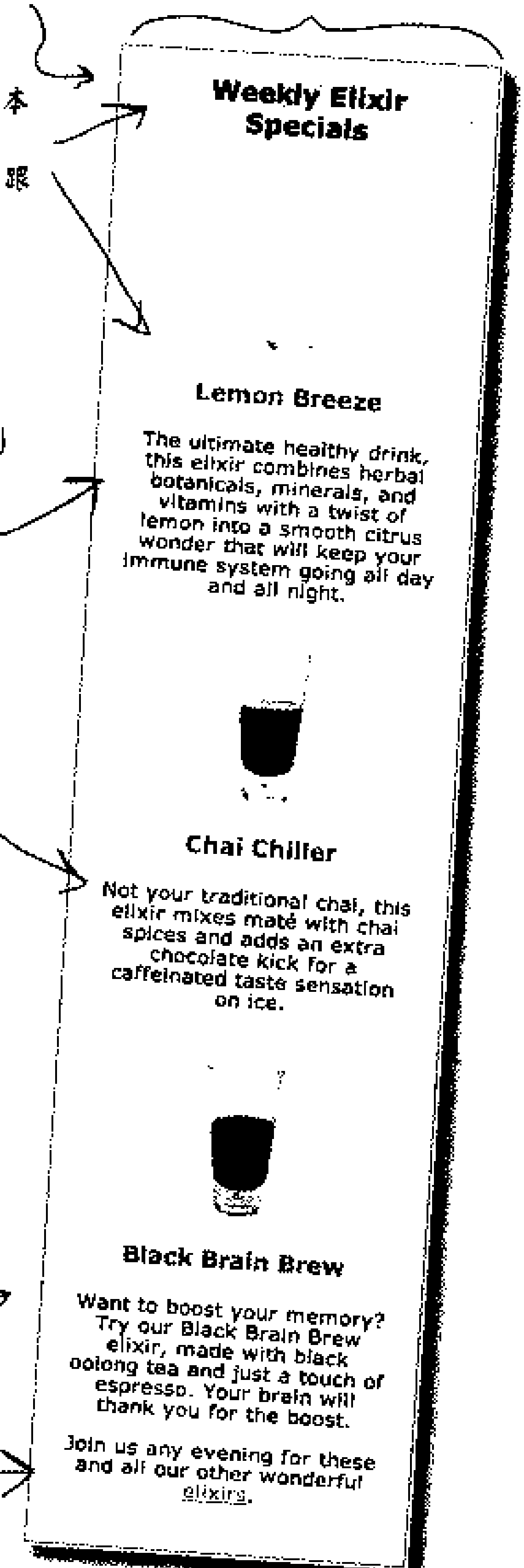
注意边框包围了<div>元素中的所有元素。<div>是一个盒子，跟其他元素差不多。所以，添加一个边框，就会包围<div>中所有的元素。



给饮料部分添加些真正的样式

到目前为止，一切还算顺利。我们已经用边框把整个部分包围起来了。以下是使用<div>单独定制整个饮料部分样式的方法。

显然还需要添加一些补白，因为边框紧挨着内容。不过还要解决很多其他样式，让我们来看一下需要注意的事项……



游戏计划

因为有许多新样式，所以处理之前先制定一个游戏计划。我们需要做的是：

- ❑ 首先，要把饮料`<div>`的宽度变窄一些。
- ❑ 接着，要快速完成已经很熟悉的一些样式，如补白和背景图像，也要设置以前没有接触过的文本对齐。
- ❑ 剩下的就是文本行间距和标题颜色了。你会发现只需提高一点CSS选择器技能就可以完成这些改动。

有很多事要做，那我们就开始吧。

调整饮料部分宽度

我们希望饮料部分窄一些，看上去就跟休闲室散发的传单一样。宽度最好是典型的浏览器窗口宽度的1/4左右。一般浏览器窗口宽度为800像素，1/4就是200像素。你设置过补白、边框和边界的宽度，但还没设置过元素的宽度，这要用**width**属性，就像这样：

```
#elixirs {
        border-width:  thin;
        border-style:  solid;
        border-color:  #007e7e;
        width:  200px;
}
```

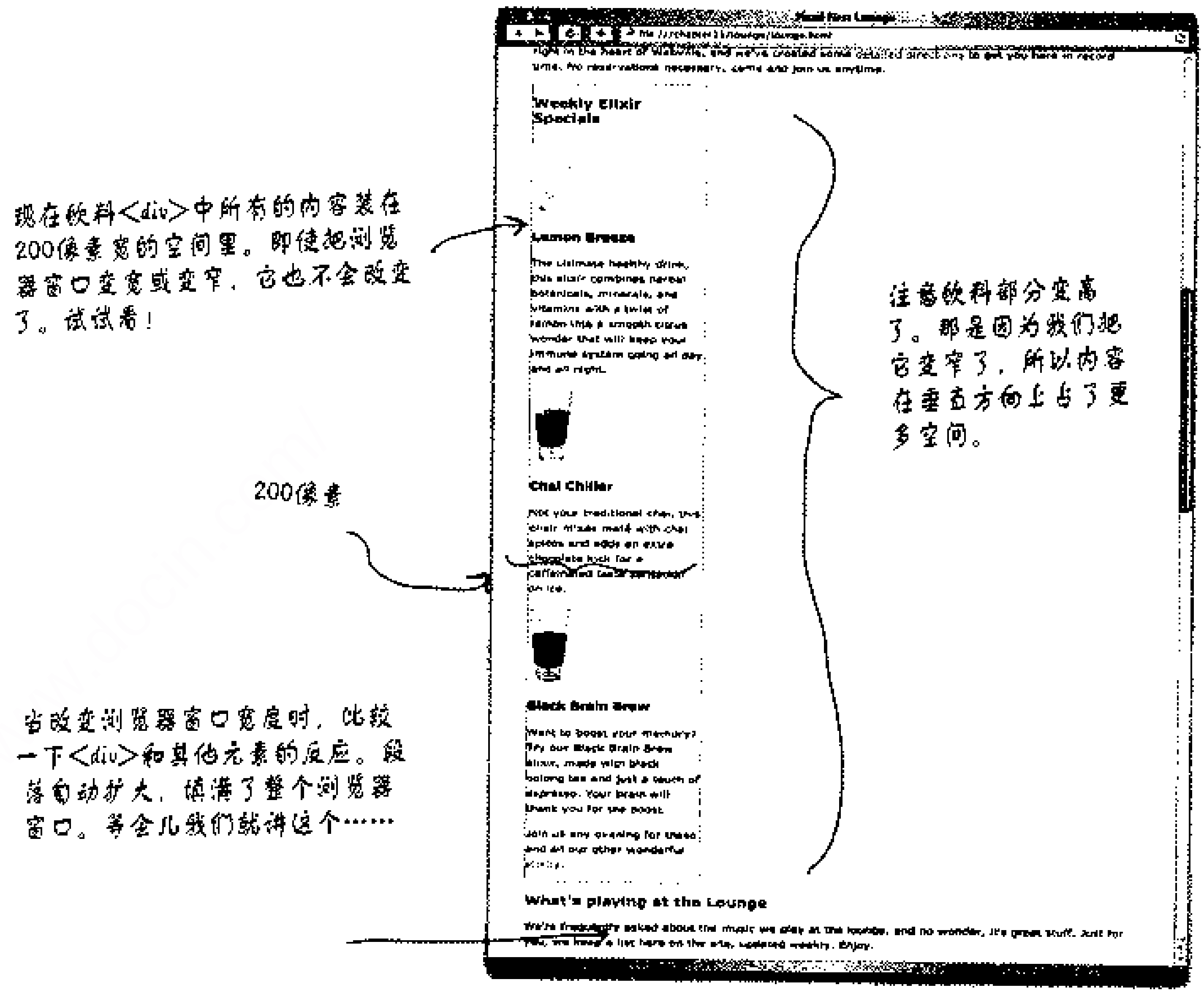
width属性用来定义元素的内容区的宽度。这儿我们把内容区宽度定义为200像素。

我们把饮料<div>设置成这样。所以饮料<div>中内容的宽度将是200像素，浏览器布局规则会把嵌套在<div>中的所有元素调整成这个宽度。

试一试，打开“lounge.css”并把这条规则添加到末尾。

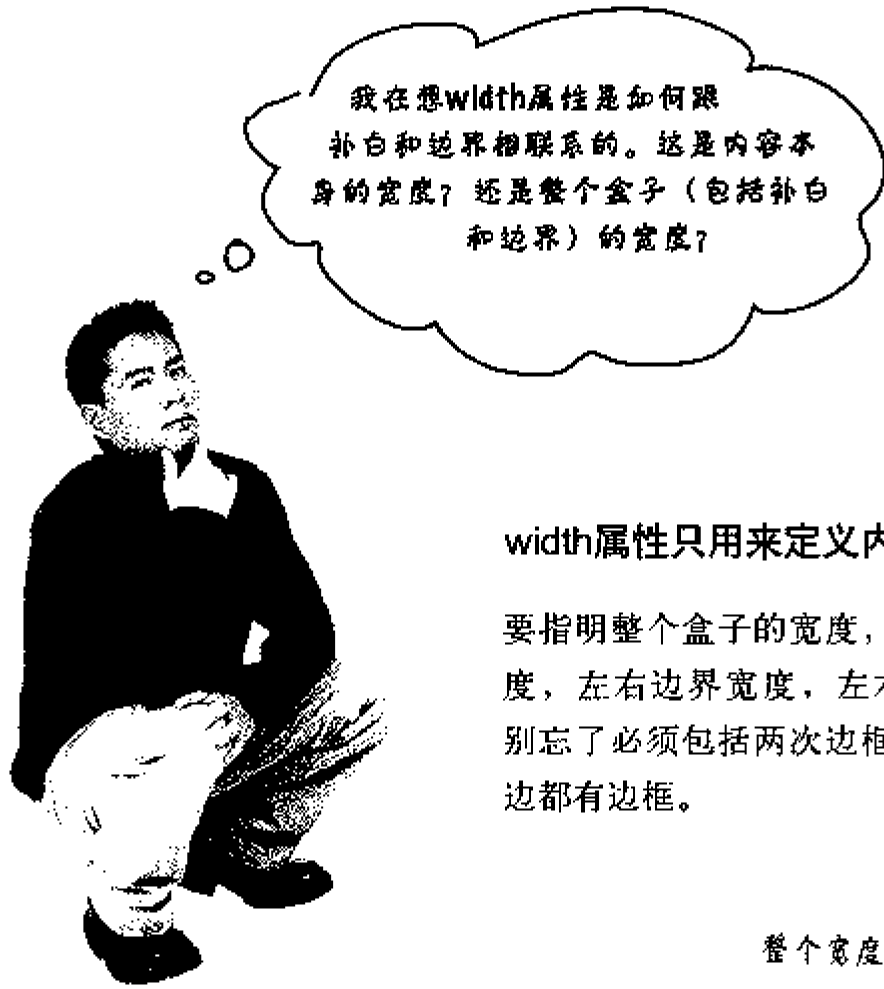
测试宽度

接下来，保存CSS，重新载入“lounge.html”文件。你会看到饮料部分变得很窄，这取决于设置的宽度。现在**<div>**中内容的宽度正好是200像素。还有很多有趣的结果，你应该仔细观察一下……



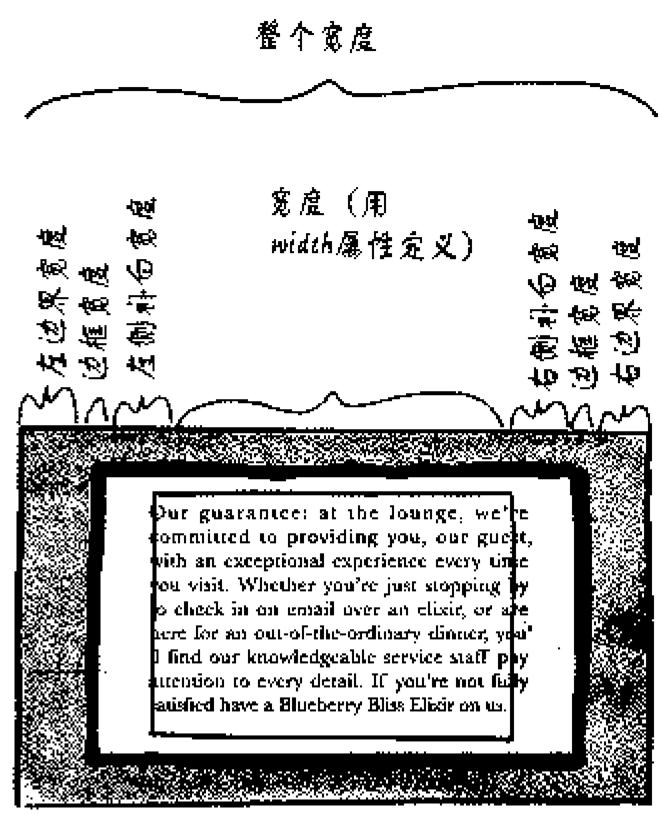
BRAIN POWER

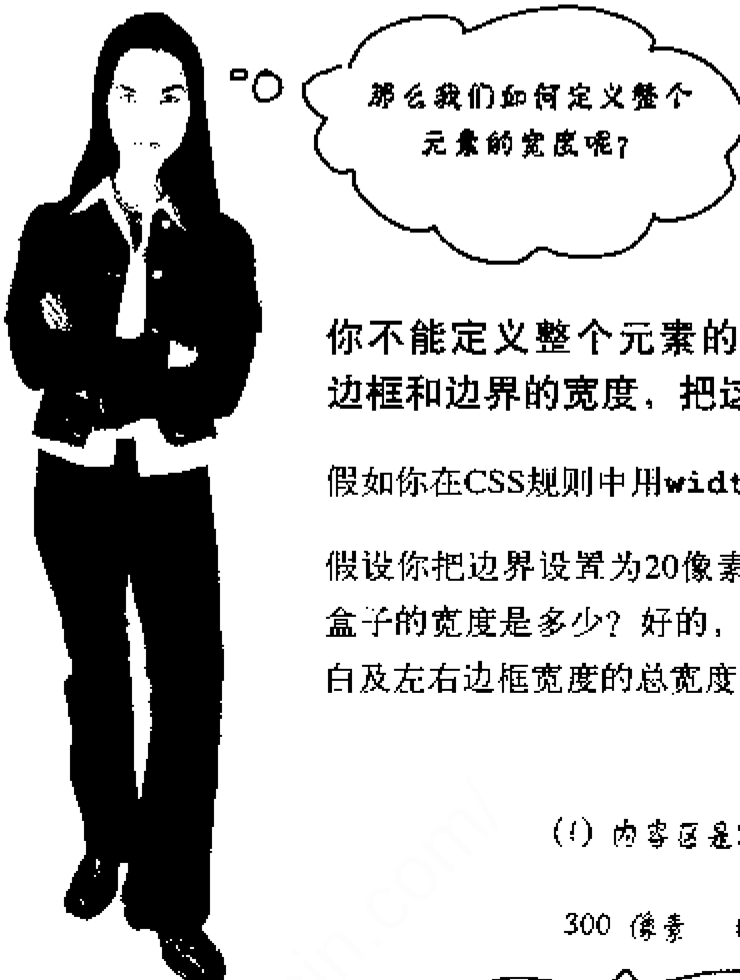
你能把浏览器窗口宽度调整到比饮料<div>还窄吗？有些浏览器不能调到那么窄，有些能。如果能，比较饮料<div>中的文本和页面中的其他文本。无论你调到多宽或多窄，别的段落会自己调整，但是饮料<div>的宽度总是200像素。



width属性只用来定义内容区的宽度。

要指明整个盒子的宽度，需要添加内容区的宽度，左右边界宽度，左右侧补白及边框宽度。别忘了必须包括两次边框宽度，因为左边和右边都有边框。

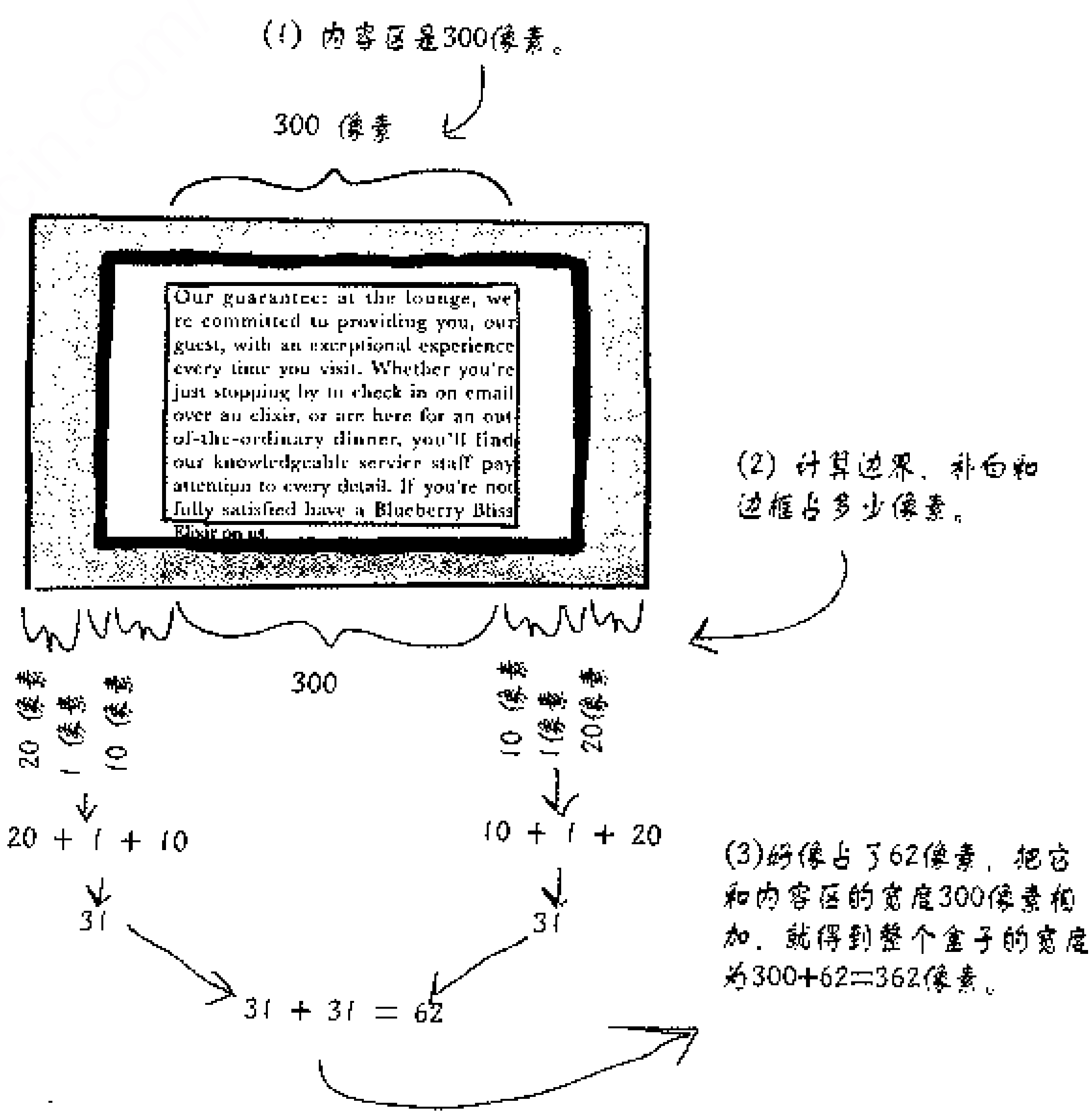




你不能定义整个元素的宽度。你只能定义内容区、补白、边框和边界的宽度，把这些全加起来就是整个元素的宽度。

假如你在CSS规则中用`width`属性把内容区的宽度定义为300像素。

假设你把边界设置为20像素，补白为10像素，边框为1像素，那么元素盒子的宽度是多少？好的，是内容区宽度加上左右边界宽度、左右侧补白及左右边框宽度的总宽度。看一下如何计算……



there are no Dumb Questions

问： 如果不设置元素的宽度，宽度是怎么得来的？

答： 块元素的默认宽度是“auto”，就是说会延伸到所有的空间。想一下我们做过的任何一个网页，每个块元素都可以延伸到浏览器的边界，就是因为“auto”。我们将在下章详细讲解这一点。记住“auto”使内容充满所有的空间（除了补白、边框和边界）。

问： 如果没有边界、补白或边框呢？

答： 那么内容就会占满整个盒子。如果内容区的宽度是300像素，而没有补白、边框或边界，整个盒子的宽度也就是300像素。

问： 还有什么方法可以用来定义宽度吗？

答： 可以定义实际大小——通常用像素——或者也可以定义百分数。如果用百分数，宽度就是相对于元素所在的容器（可以是<body>、<div>等）的宽度的百分比。

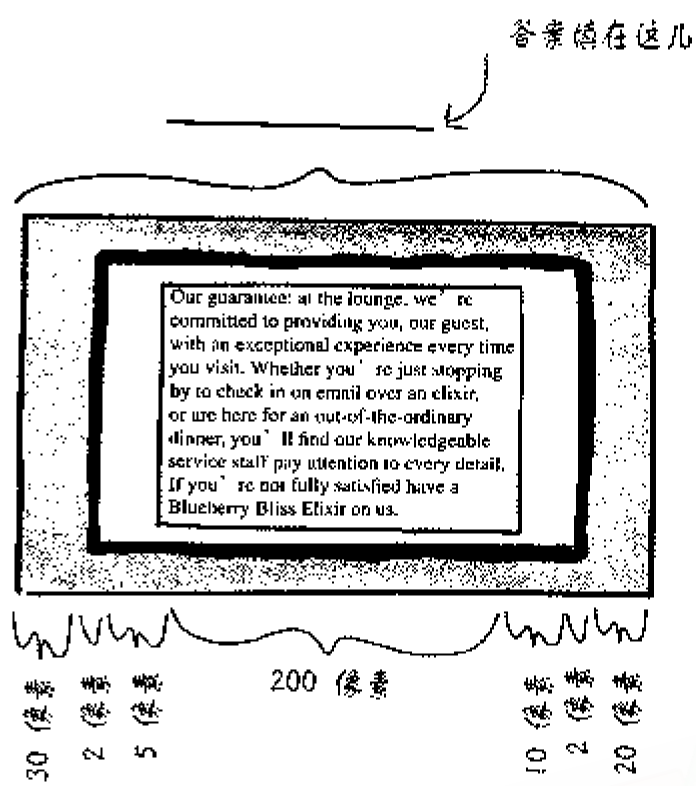
问： 高度怎么定义呢？

答： 一般元素的高度用默认值，就是auto，浏览器垂直扩展内容区，以使所有的内容都能看到。我们把宽度设置为200像素后，看一下饮料部分，你会发现<div>变得高多了。

可以明确地设置高度，但是如果设置的高度太小，内容尾部有可能丢失。因此一般不用定义元素高度，而默认为auto。我们将在下一章介绍更多关于这方面的内容。

Sharpen your pencil

以下是一个标出了所有宽度的盒子，整个盒子的宽度是什么？



给饮料添加基本样式

宽度已经解决了，还有什么要做呢？

- ☑ 首先，要把饮料<div>的宽度变窄一些。
- ☐ 接着，要快速完成已经很熟悉的一些样式，如补白和背景图像，还要设置以前没有接触过的文本对齐。（马上要做这一步。）
- ☐ 剩下的就是文本行间距和标题颜色。你会发现只需提高一点CSS选择器技能就可以完成这些改动。

现在我们将专注于一些基本样式，比如补白、文本对齐，还要在饮料<div>中添加鸡尾酒杯的背景图像。你已经熟悉大多数工作了，我们快速浏览一下CSS：

记住我们只把这个样式应用到饮料<div>，所以它只对<div>和它包含的元素起作用，而不是整个页面。

```
#elixirs {
    border-width:      thin;
    border-style:      solid;
    border-color:      #007e7e;
    width:             200px;

    padding-right:     20px;
    padding-bottom:    20px;
    padding-left:      20px;

    margin-left:       20px;

    text-align:        center;

    background-image:  url(images/cocktail.gif);
    background-repeat: repeat-x;
}
```

默认的<div>的补白是0像素，所以要添加一些补白以增加内容的空间。注意上面没添加补白，这多亏<h2>标题默认的边界已经有足够的空间（回顾上一个测试，你会发现<h2>上面有很多空间），但是左侧、右侧和下面需要有补白。

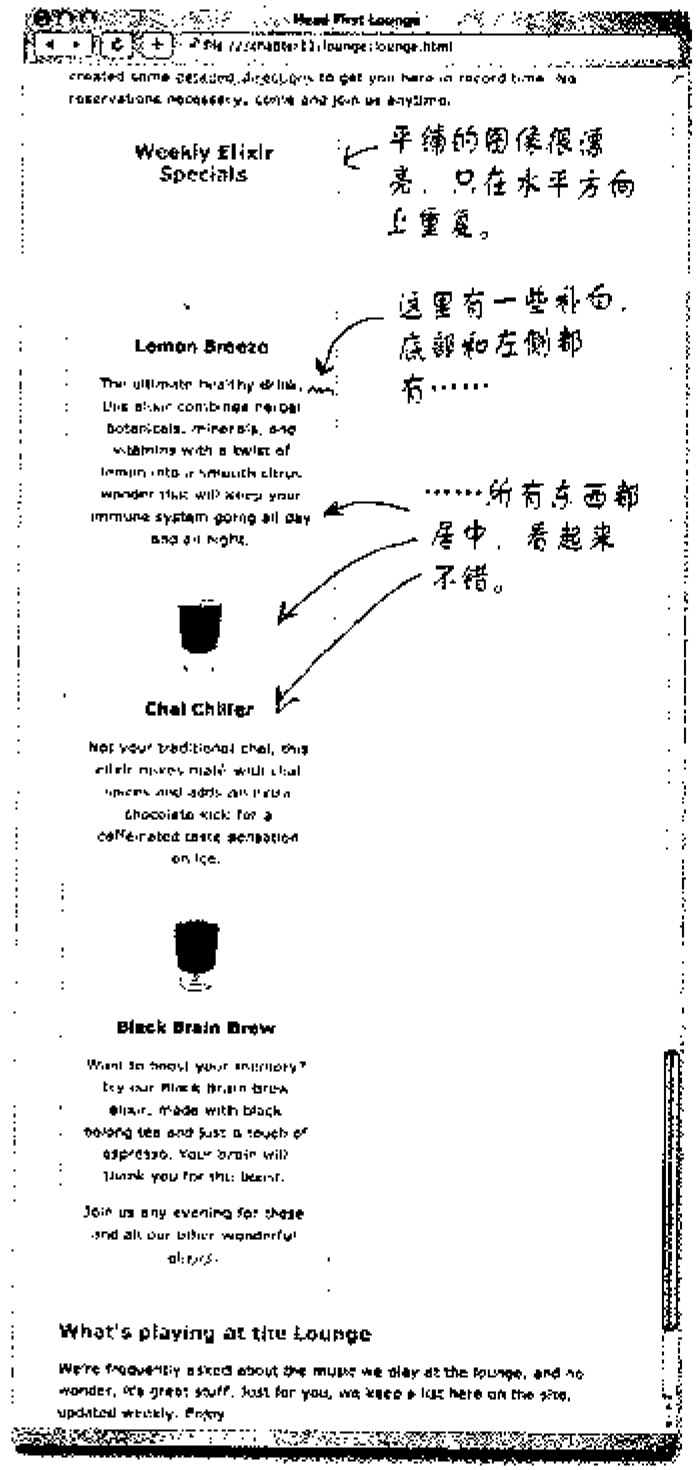
在左侧加了一些边界，使饮料部分比页面其他部分稍微有些缩进。以后就会有用……

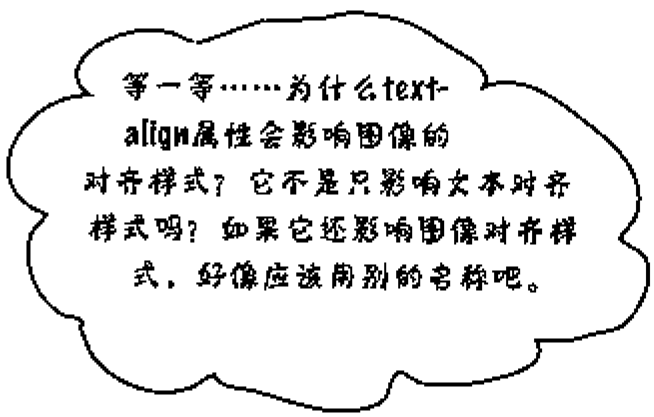
在块元素中用text-align来设置其中的文本对齐样式。这儿设置为居中。

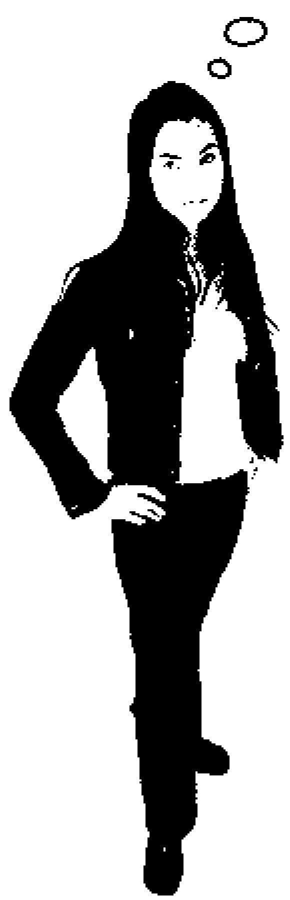
最后定义一个图像作为背景，在这个例子中是鸡尾酒杯的图像。我们把background-repeat属性设置为repeat-x，只在水平方向上重复图像。

测试新样式

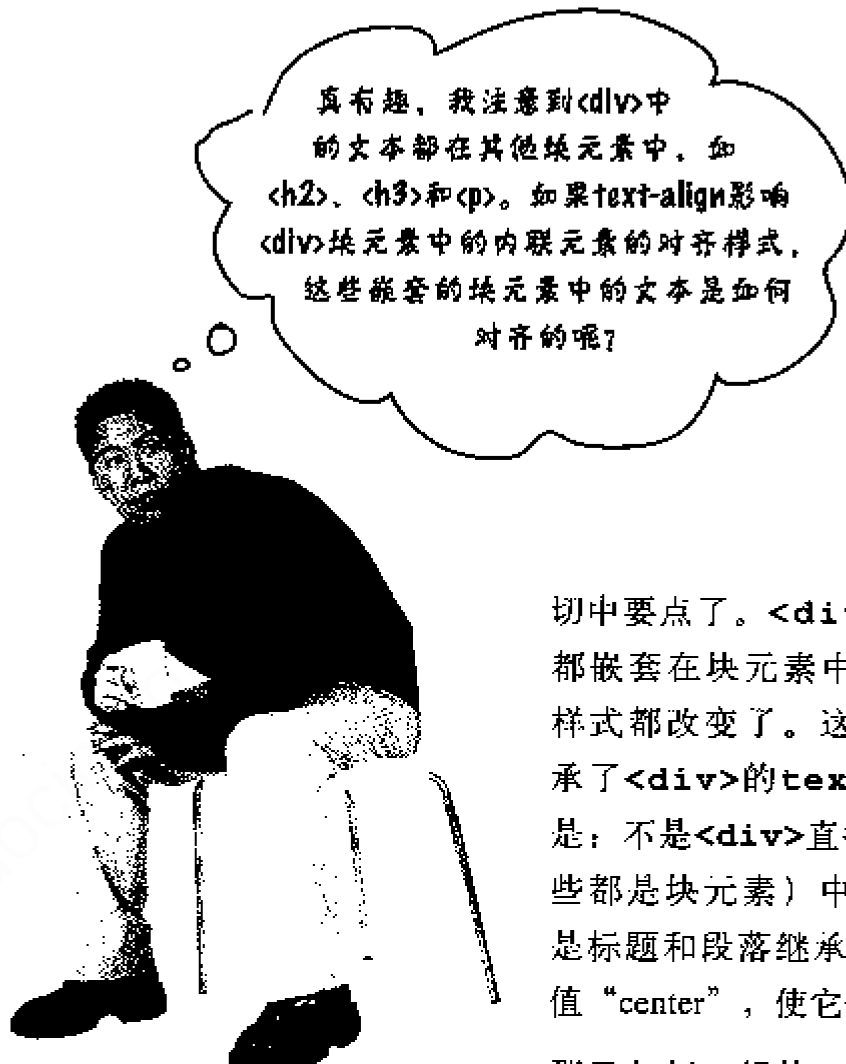
现在把这些新属性添加到你的“lounge.css”文件中，并重新载入页面。仔细观察一下变化：**<div>**中的标题，图像和文本都居中了，并且因为添加了一些补白有了更多的呼吸空间。在顶部还添加了平铺的鸡尾酒杯图像作为装饰。







这次说到了点子上……好像不对，是吗？其实text-align属性会影响一个元素中所有内联内容的对齐样式。在这个例子中，我们在**<div>**块元素中设置了这个属性，结果它的所有内容都居中了。记住不管**text-align**的名字叫什么，都会影响各种内联元素。还要记住，**text-align**属性只能用于块元素，如果直接用于内联元素（如**<img>**）就没有作用了。



切中要点了。`<div>`元素中的所有文本都嵌套在块元素中，但现在它们的对齐样式都改变了。这是因为这些块元素继承了`<div>`的`text-align`属性。区别是：不是`<div>`直接影响标题和段落（这些都是块元素）中的文本对齐样式，而是标题和段落继承了`text-align`属性值“center”，使它们自己的内容居中了。

那又如何？好的，如果你好好考虑一下，就会有很多方法来使用`<div>`，因为可以用`<div>`包围一部分内容，然后把样式应用于`<div>`而不是每个单独的元素。当然，谨记并非所有的属性都是可以默认继承的，所以这并不会对所有的属性都起作用。

Sharpen your pencil

现在你已经理解了宽度，那么饮料盒子的总宽度是多少呢？首先，内容区是200像素，宽度还在左侧和右侧设置了补白，它们也会影响宽度，还有一个设置为“thin”的边框。在大多数浏览器中，thin边框宽度是1像素宽。边界呢？我们只设置了一个左边界值，没有右边界值，所以默认的右边界是0像素。

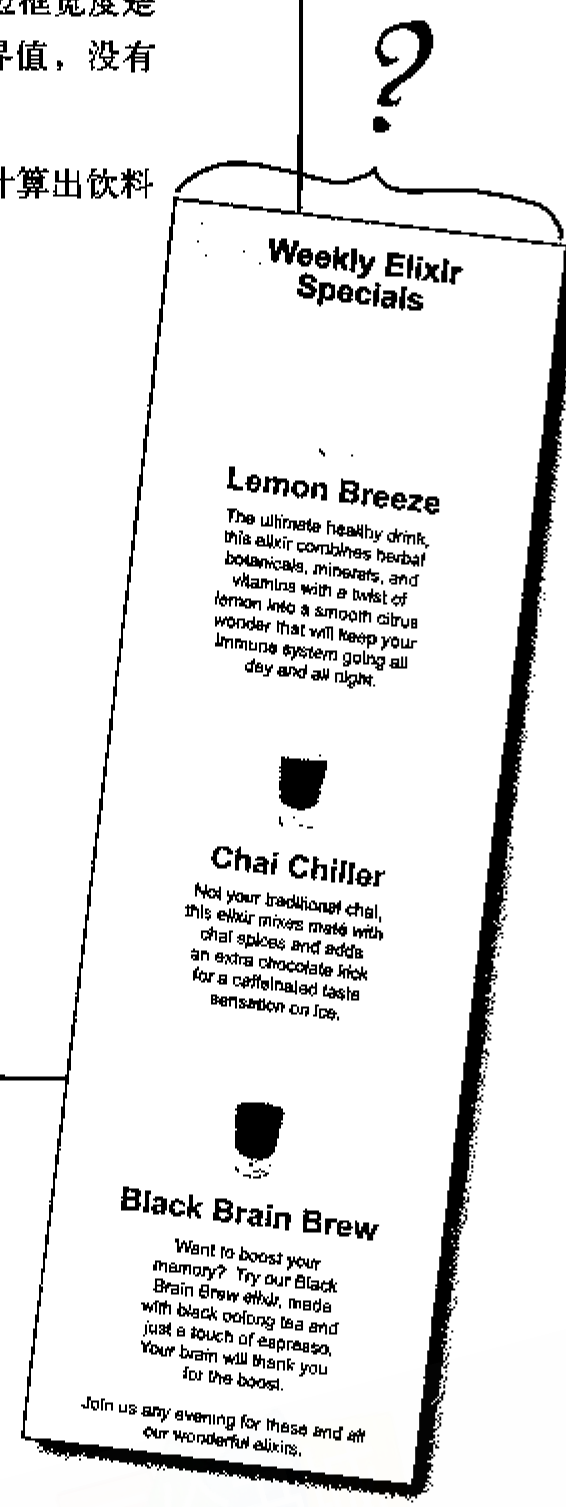
以下是所有跟宽度有关的属性。你的工作是计算出饮料<div>的总宽度。

```
border-width:    thin;

width:           200px;

padding-right:   20px;
padding-bottom:  20px;
padding-left:    20px;

margin-left:     20px;
```



快要完成了……

我们差不多要完成饮料部分了。还剩什么？

- ☑ 首先，要把饮料<div>变窄一些。
- ☑ 接着，要快速完成已经很熟悉的一些样式，比如补白和背景图像。还得设置以前没有接触过的文本对齐。
- ☐ 剩下的就是文本行间距和标题颜色。你会发现只需提高一点CSS选择器技能就可以完成这些改动。 ← 只剩这最后一步了。

听起来很简单，对吗？毕竟，你以前做过这些。假如你知道可以只设置<div>的样式，而且它们会被继承，那么你就可以很快地完成这些任务了。



Frank：耶！太有趣了，饮料的主标题<h2>是浅绿色，因为CSS中已经有一条<h2>规则了。但我们要把它变成黑色，还要把饮料中的<h3>变成红色。

Jim：对，没问题，只要再添加一些规则就行了。

Frank：不过等一等……如果改变<h2>规则或添加一条<h3>规则，就会改变整个页面的标题颜色。可我们只想改变饮料部分的标题颜色呀。

Jim：嗯，说得好。嗯……好的，我们可以用两个类。

Frank：尽管有些乱，也还行。只是你往饮料<div>添加一个新标题一定要记得把它加进类中。

Jim：好的,这就是生活。

Frank：其实在用类之前，先考察一下子孙选择符。我觉得这儿用它们会更好一些。

Jim：子孙选择符？

Frank：对，用它们可以定义一个这样的选择符——比如“选择一个饮料<div>中的<h2>元素”。

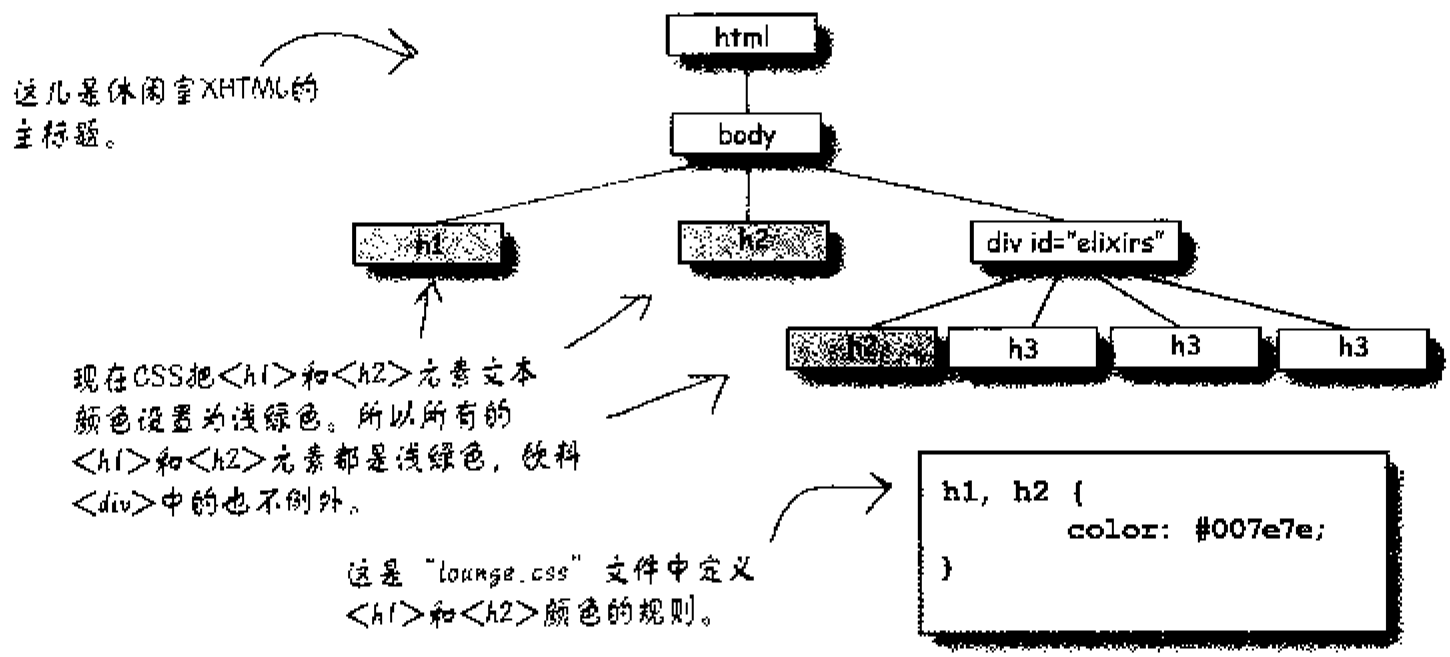
Joe：我不是很懂。

Frank：没关系，我们一步一步来……

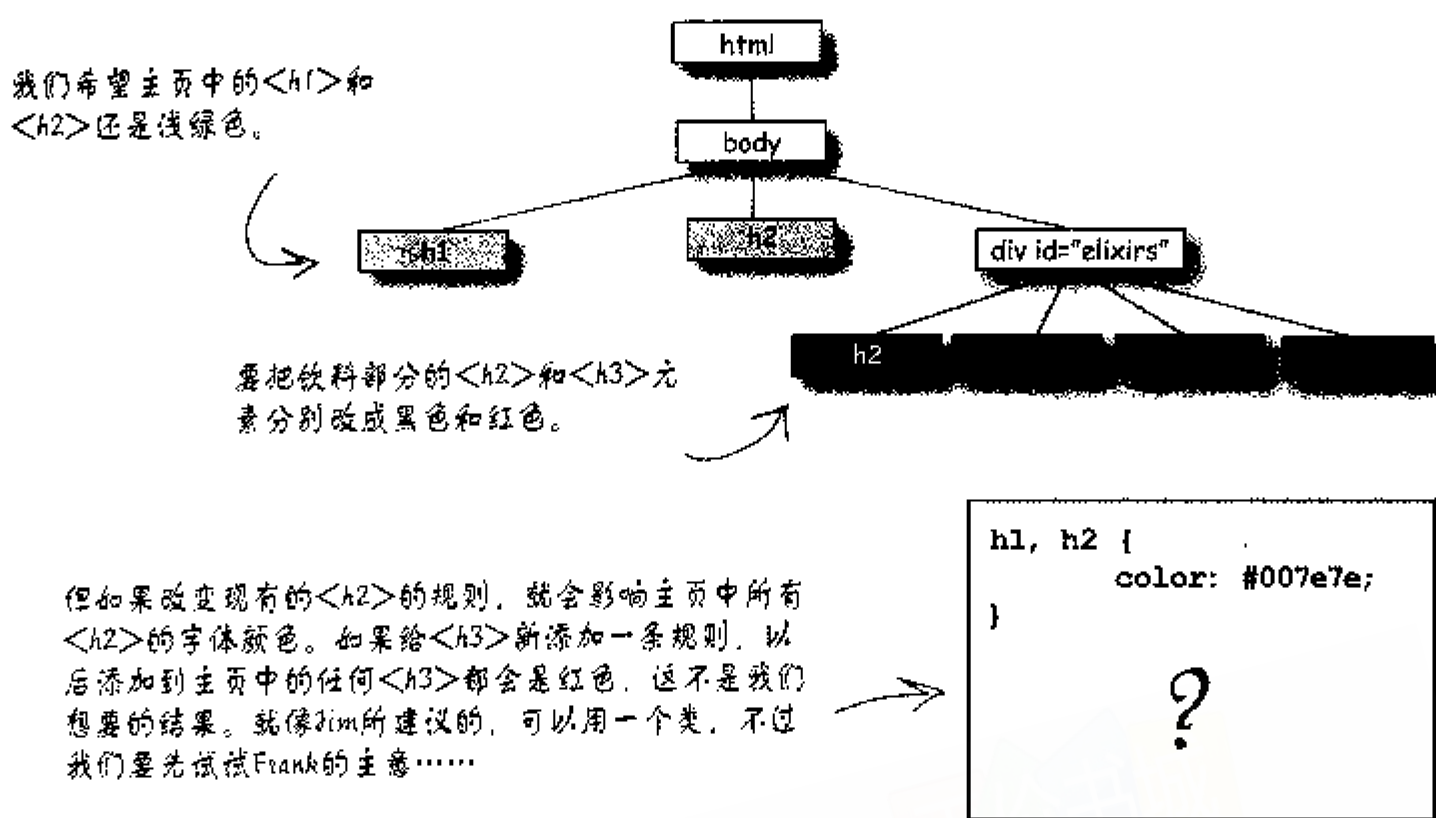
我们的目标是什么？

快速看一下我们对标题颜色有什么要求。

我们现在有什么



我们想要什么



我们需要一种选择子孙的方法

需要一种告诉CSS我们只想选择特定元素的子孙元素的方法，有点像指定你的遗产只能由某个女儿或儿子继承。写一个子孙选择符的方法如下：

父名和子孙名之间有空格。

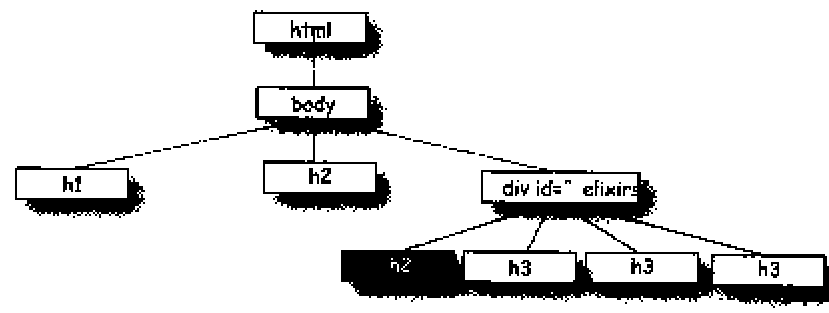
这是父元素。

这是它的子孙

```
div h2 {
    color: black;
}
```

规则的其余部分跟平常写的一样。

这条规则表明选择<div>的子孙<h2>。



这是这条规则在休闲室中选择的内容。

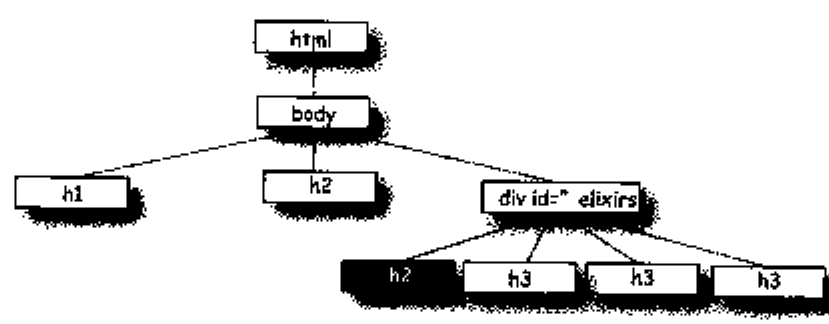
现在这条规则唯一的问题是如果有人在“lounge.html”文件中创建了另外一个**<div>**，他们的**<h2>**文本也会是黑色的，即使他们不希望这样。不过饮料**<div>**有一个**id**，可以用它来更具体地指出我们想选择哪个子孙：

现在父元素是id为“elixirs”的元素。

这是它的子孙。

```
#elixirs h2 {
    color: black;
}
```

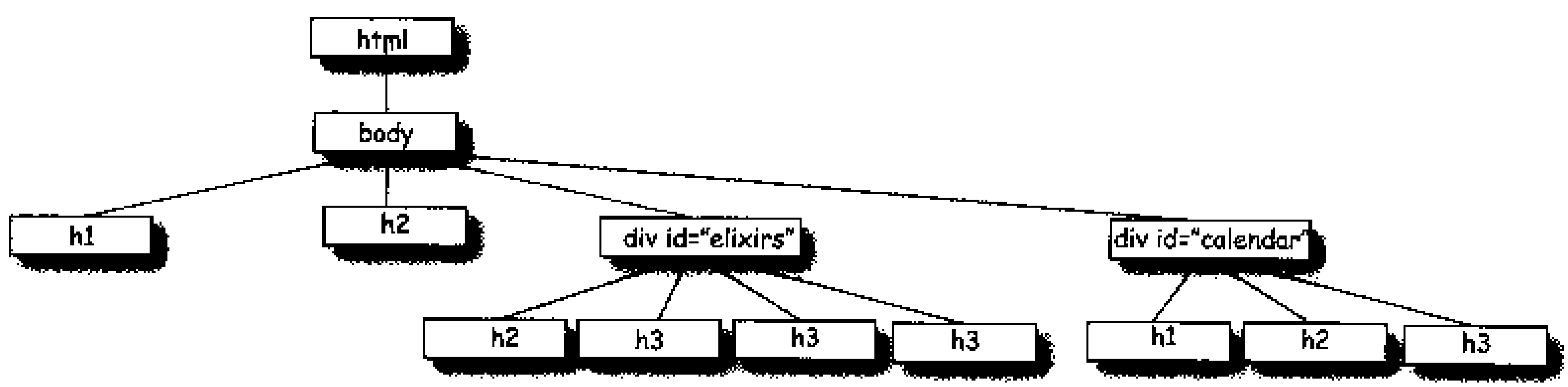
这条规则表明选择id为“elixirs”的元素的子孙<h2>。



这条规则选择了同样的元素。不过它更具体一些，如果我们给页面添加另外一个有<h2>的<div>也不会再出问题，因为这条规则只选择饮料<div>中的<h2>。

Sharpen your pencil

轮到你了。写出只选择饮料<div>中的<h3>元素的选择符。在你的规则中，把颜色属性设置为#d12c47，同时在下图中标出你选的元素。



there are no Dumb Questions

问： 子孙通常是指孩子，孙子，曾孙。这里我们只选择孩子对吗？

答： 问得真好。选择符"#elixirs h2"是要选择饮料中所有elixirs子孙，所以<h2>可以是<div>的直接孩子，也可以是嵌套在一个<blockquote>中或另一个嵌套在<div>中的孩子（孙子）等。所以子孙选择符选择任何嵌套在一个元素中的<h2>，无论它嵌套得有多深。

问： 那么，有没有只选择直接孩子的方法啊？

答： 有的。例如，用"#elixirs> h2"选择的<h2>是id为"elixirs"的元素的直接孩子。

问： 如果我需要一些更复杂的东西，比如一个<h2>，是在饮料部分中一个<blockquote>的孩子，那该怎么办呢？

答： 可以用同样的方法。只要多用一些子孙，像这样：

```
#elixirs blockquote h2 {
    color: bulue;
}
```

这就会选择一个id为"elixirs"的元素的子孙<blockquote>的子孙<h2>元素。

改变饮料标题的颜色

你已经知道了子孙选择符，我们就来把饮料部分的<h2>标题设置为黑色，<h3>设置为红色。做法如下所示：

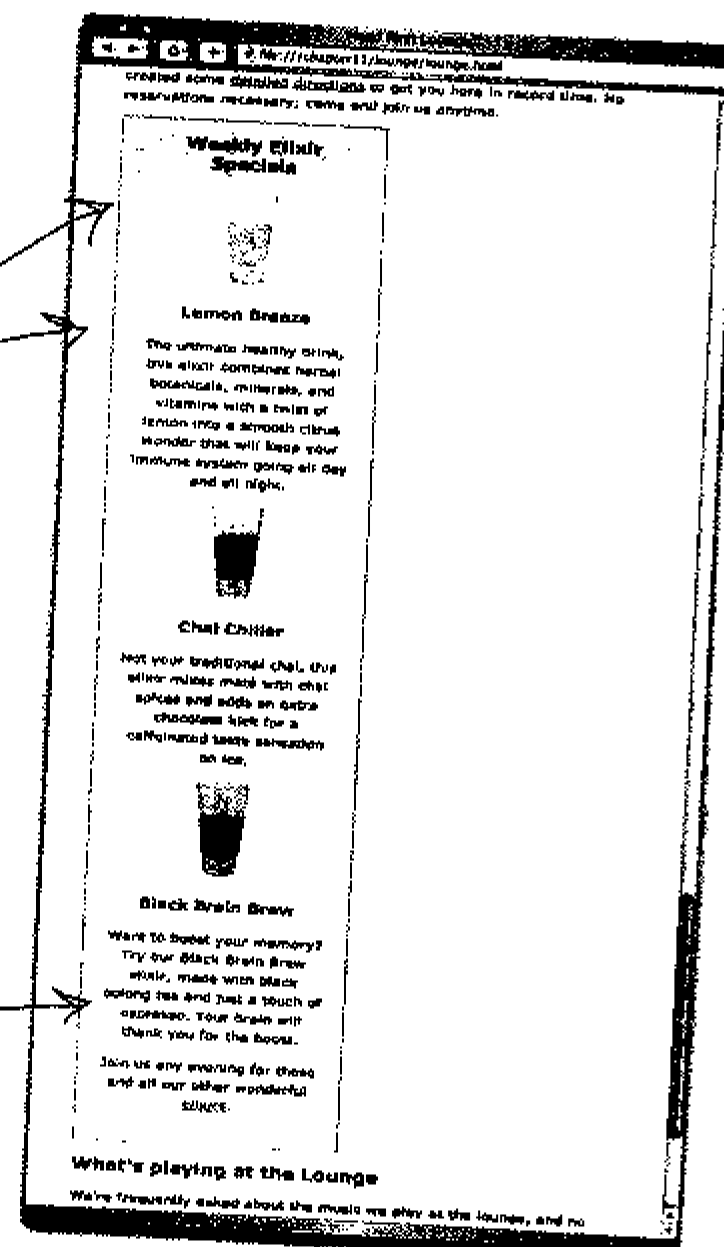
```
#elixirs h2 {
    color: black;
}

#elixirs h3 {
    color: #d12c47;
}
```

我们用子孙选择符选择饮料<div>中的<h2>和<h3>元素，用十六进制代码把<h2>设置为黑色，把<h3>设置为红色。

快速测试……

动手把这些新属性添加到“lounge.css”文件末尾，保存并重新载入“lounge.html”。



饮料部分的标题变成了黑色和红色，主页中的<h2>标题还是浅绿色。

现在只需确定行间距。

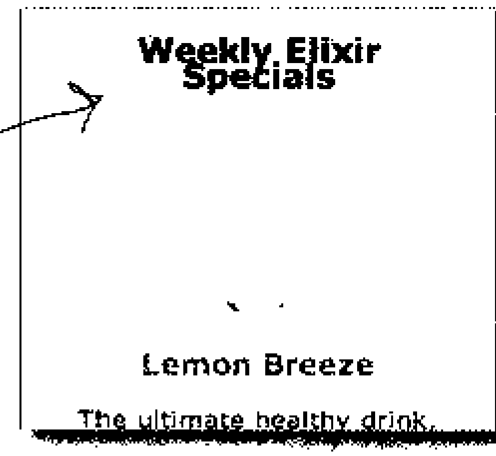
www.docin.com

确定行间距

上一章我们把休闲室文本的行间距调得比正常高了一些，看起来很不错，但是我们想让饮料部分文本的行间距都是正常的，跟传单一样。听起来十分简单，对吗？只要设置<div>的line-height属性，一切就都没问题了。因为line-height是继承的，唯一的问题是标题也会继承那个行间距。以下是末尾要添加的：

```
#elixirs {
    line-height: 1em;
}
```

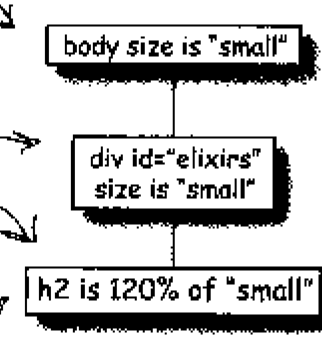
如果设置整个<div>的line-height属性，那么它就会被<div>中的所有元素继承，包括标题。注意标题的行间距太小，所以两行挤到一起了。



饮料部分标题的line-height太小是因为饮料**<div>**中的所有元素都继承了1em的line-height，即饮料元素的字体大小的1倍，饮料元素的字体大小在这个例子中是“small”，大约是12像素（取决于浏览器）。记住，饮料**<div>**继承了**<body>**元素的font-size，**<body>**元素的font-size是“small”。

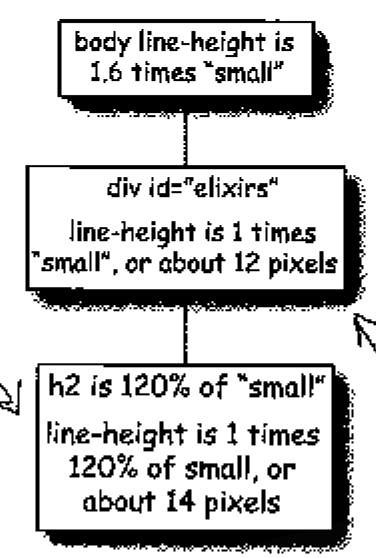
这是元素的字体大小。body字体大小设置为“small”，所以饮料部分继承了它。

把<h2>的line-height设置成了饮料部分字体大小（即为“small”，大约为12像素）的1倍。



饮料**<div>**中的所有元素的行间距都基于饮料**<div>**的font-size，我们不希望这样，最好每个元素的行间距都基于自己的font-size。我们希望**<h2>**标题的line-height是它自己的font-size的1倍（即“small”的120%），**<p>**的line-height也应该是它自己的font-size的1倍（即“small”）。如何做到这点呢？line-height有些特殊，可以只用一个数字代替相对值（比如em或%）来设置line-height的值。如果只用一个数字1，就是指每个饮料**<div>**中的元素的行间距为它们自己的font-size的1倍，而不是饮料**<div>**的font-size的1倍。试一试，把饮料**<div>**的line-height设置为1，你会发现标题的问题解决了。

我们希望<h2>元素的line-height是它自己字体大小的1倍，字体大小是14像素（small的120%）。




```
#elixirs {
    line-height: 1;
}
```

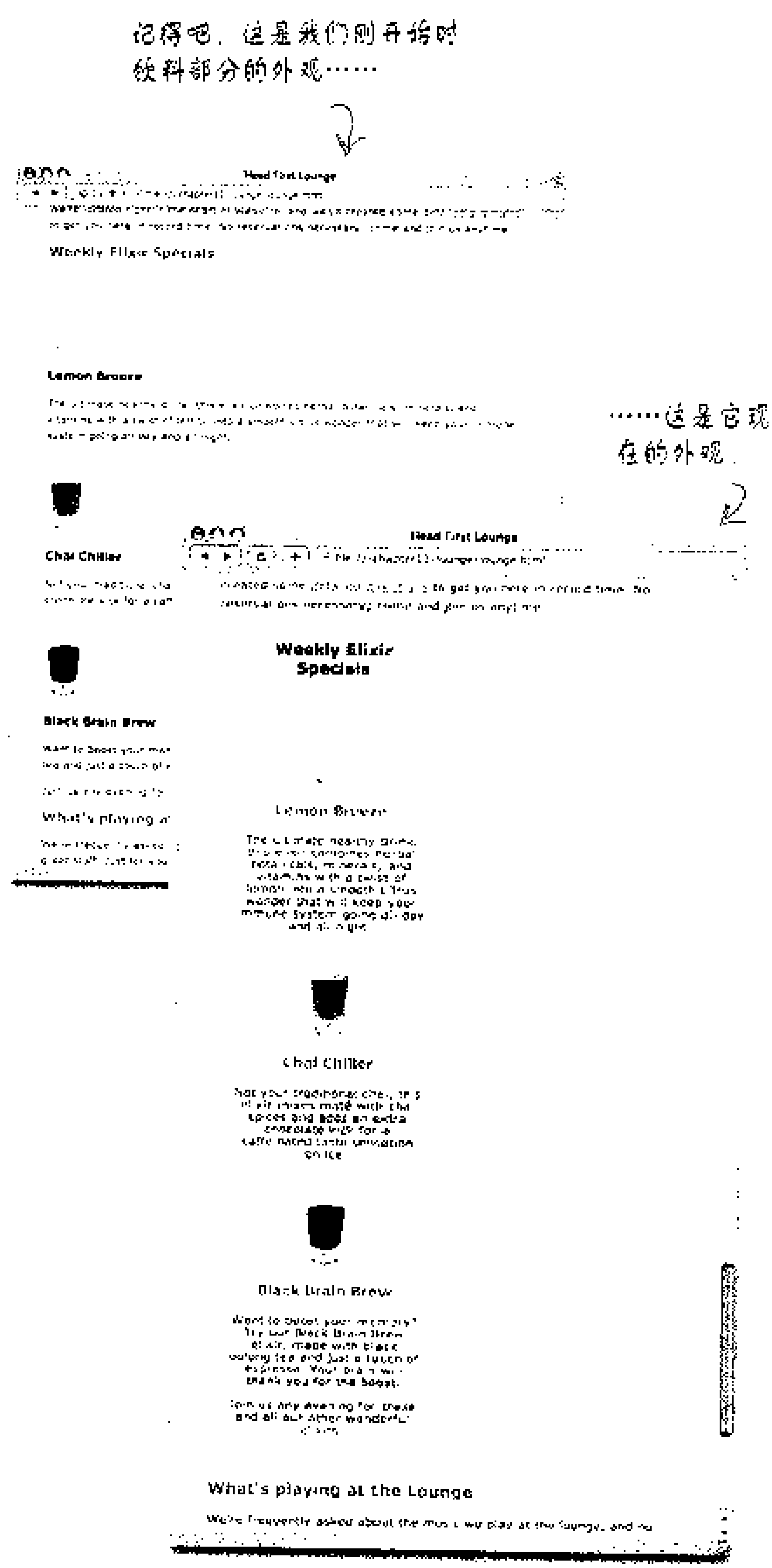
给饮料<div>添加值为1的line-height来改变它里面的每个元素的line-height。

<p>元素的font-size是“small”（<p>继承了饮料<div>的font-size），所以它的line-height就是12像素，正是我们想要的。

已经完成了什么……

现在看一下饮料部分，它已经完全改变了，现在看起来跟传单一样。仅仅给XHTML添加一个<div>和一个id属性，再用几行CSS规则和属性就可以做到这些。

现在，你应该认识到CSS有多强大，结构（XHTML）和外表（CSS）分开时网页有多灵活了。可以通过只改变CSS让XHTML有一个全新的外观。





走点儿捷径

也许你已经注意到，许多CSS属性要一起使用，例如，**padding-left**，**padding-right**，**padding-bottom**和**padding-top**。边界属性也一样。**Background-image**，**background-color**和**background-repeat**也是吗？这些用来设置元素背景的不同属性值，难道你不觉得把它们全输进去有些啰嗦吗？把这些全输进去的时间可以用来做更好的事情，对吗？

```
padding-top:    0px;
padding-right:  20px;
padding-bottom: 30px;
padding-left:   10px;
```

为了指定这四个数字要打很多字。

本章有一些捷径来解决这个问题。你就再也不用冒着得腕骨综合症的危险指定这些值了，方法如下：

这是指定补白的老办法。

```
padding-top:    0px;
padding-right:  20px;
padding-bottom: 30px;
padding-left:   10px;
```

这是新的改进了的写成缩写的方法。

```
padding: 0px 20px 30px 10px;
```

上 右 下 左

跟补白一样，也可以用缩写指定所有的边界值：

```
margin-top:    0px;
margin-right:  20px;
margin-bottom: 30px;
margin-left:   10px;
```

```
margin: 0px 20px 30px 10px;
```

上 右 下 左

跟补白一样，也可以用缩写指定所有的边界值。

如果各侧面的补白和边界值都相等，缩写就会更简单：

```
padding-top:    20px;
padding-right:  20px;
padding-bottom: 20px;
padding-left:   20px;
```

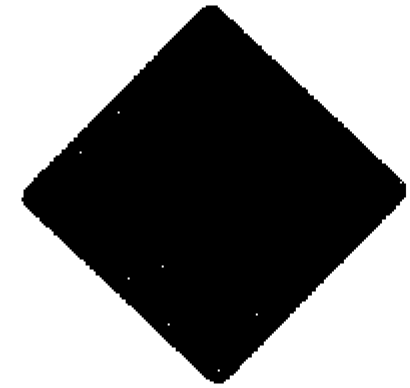
```
padding: 20px;
```

这表明盒子四周的补白都是20像素。

如果补白值都相等，可以这么写。

还有很多……

还有一个常见的缩写边界的方法：



```
margin-top:     0px;
margin-right:   20px;
margin-bottom:  0px;
margin-left:    20px;
```

上下边界相同。

左右边界相同。

如果上下边界相同，左右边界相同，可以这么缩写。

```
margin: 0px 20px;
```

上和下

右和左

边框属性又如何呢？它们同样可以缩写。

```
border-width:    thin;
border-style:    solid;
border-color:    #007e7e;
```

把边框属性都写成一个属性，顺序不限。

```
border: thin solid #007e7e;
```

边框的缩写比边界和补白更灵活，因为顺序随你选择。

这些都是完全有效的边框缩写。

```
border: solid thin #007e7e;
border: #007e7e solid thin;
border: solid thin;
border: #007e7e solid;
border: solid;
```

……别忘了背景的缩写

背景也可以缩写：

```
background-color:   white;
background-image:   url(images/cocktail.gif);
background-repeat:  repeat-x;
```

跟边框一样，背景属性值的缩写顺序不限。还有另外一些值也可以在缩写形式中指定，比如background-position。

```
background: white url(images/cocktail.gif) repeat-x;
```

更多缩写

讲到缩写，就一定要讲字体缩写。查看一下有关字体的所有属性：（**font-family**，**font-style**，**font-weight**，**font-size**，**font-variant**，）别忘了还有**line-height**。缩写可以把这些都包括进去，方法如下：

这是组成字体缩写的几个属性，要注意出现的顺序。

必须定义字体大小。

最后，需要添加字体系列。你只需要指定一种字体，但强烈建议指定两种以上。

```
font: font-style font-variant font-weight font-size/line-height font-family
```

这些值都是可选的。可以把它们任意组合，但要在font-size之前。

line-height是可选的，定义行间距的方法是，在font-size右边加一个/，再加属性值。

字体系列名之间加逗号。

那么让我们来试试。以下是休闲室body的字体属性：

```
font-size: small;
font-family: Verdana, Helvetica, Arial, sans-serif;
line-height: 1.6em;
```

与缩写格式的映射关系：

这些当中的任何一个都没用到，不过没关系，它们都是可选的。

```
font: font-style font-variant font-weight font-size/line-height font-family
```

缩写：

```
font: small/1.6em Verdana, Helvetica, Arial, sans-serif;
```

这是缩写版本，非常简略，不是吗？你现在有双倍的休息时间了。

there are no Dumb Questions

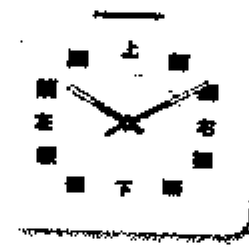
问： 一定要用缩写吗？

答： 不必。有些人觉得完整形式更方便阅读。缩写对减小CSS文件的确有好处，而且因为要打的字比较少，输入就会快一些。然而，当有问题的时候——如果值或顺序搞错了——缩写会比较难“调试”。所以，可以根据自己的喜好采用较合适的一种形式，它们都完全有效。

问： 我觉得缩写更复杂，因为我必须记住顺序，以及什么是可选的，什么是不可选的。如何才能记住这些呢？

答： 熟能生巧！但是身处“商界”的我们有一个小秘密，我们喜欢称之为“参考手册”。选好一本手册，需要时立即查阅。我们特别推荐Eric Meyer的CSS袖珍参考书。此书不仅小巧而且很有参考价值。

强化记忆



要记住补白和边界缩写值的顺序，想一下一个标记了上、右、下、左的钟，总是以指针走动的方向写：从上到右再到下再到左。

```
margin: 0px 20px 30px 10px;
```

练习

该把你的新知识付诸实践了。休闲室的末尾有一小部分是版权信息，用它作为页面的页脚。加一个<div>，组成它自己的逻辑部分。完成后用以下属性给它设计样式：

```
font-size: 50%;
text-align: center;
line-height: normal;
margin-top: 30px;
```

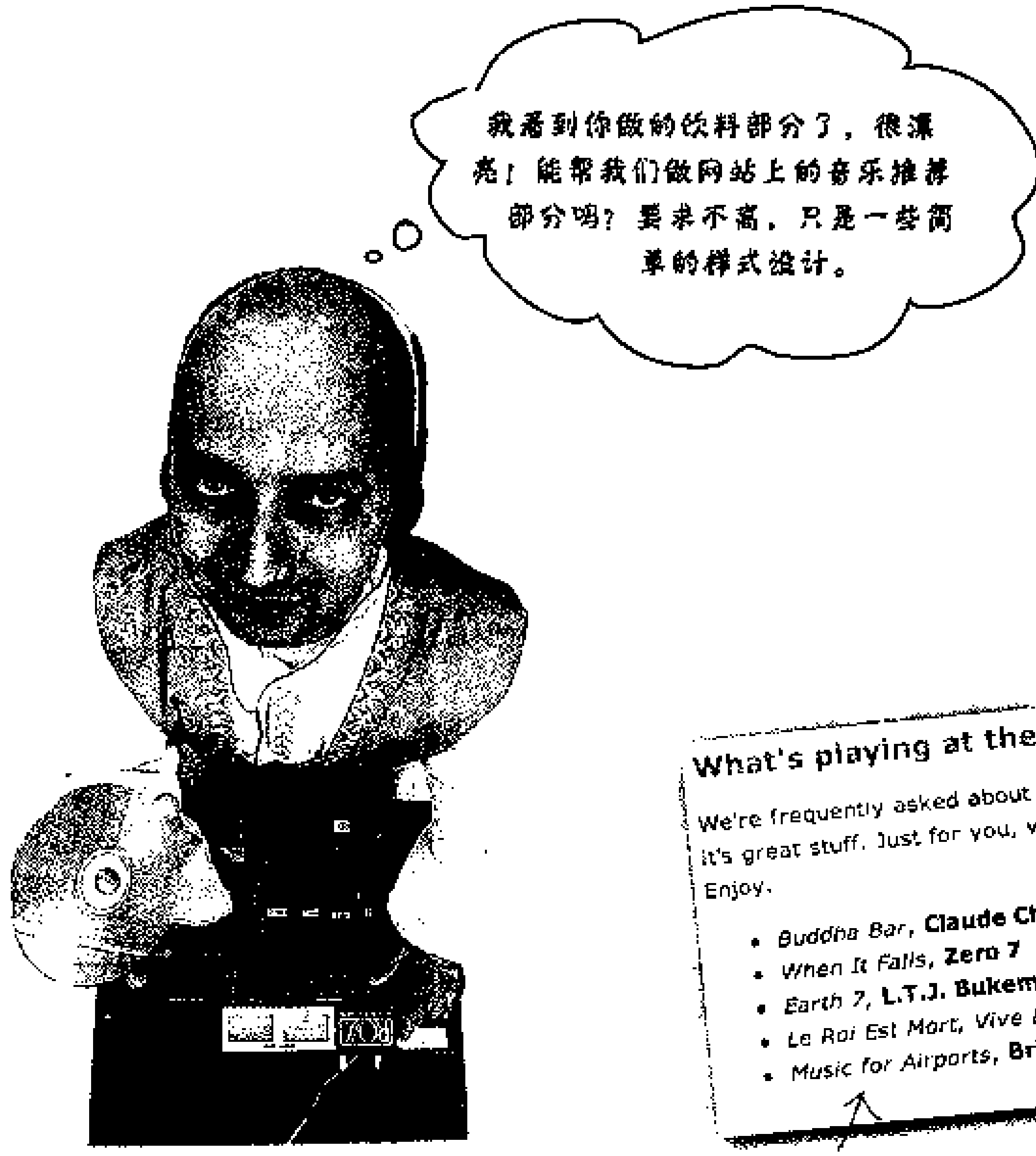
文本做成极小的字体。

文本居中。

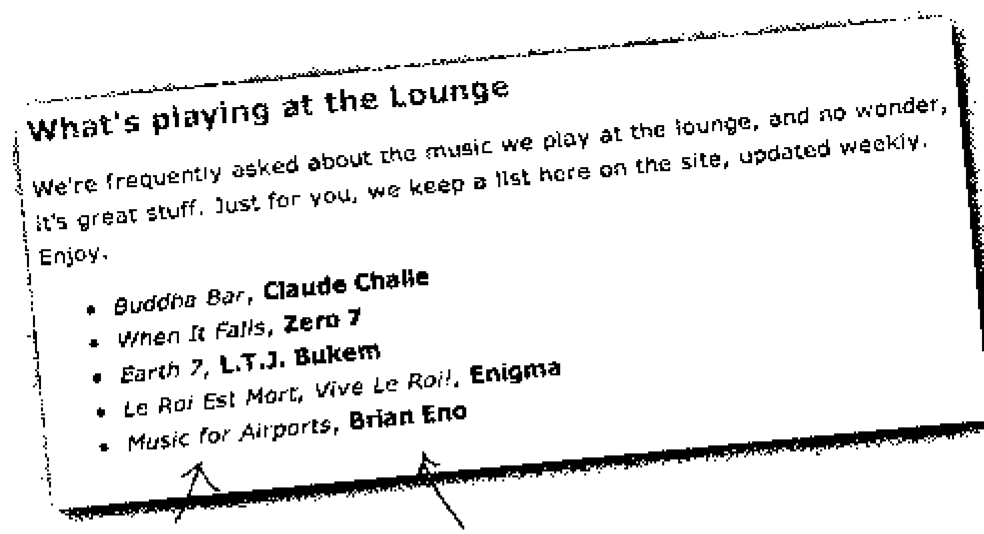
把line-height设置成“normal”，这是一个你没见过的关键字。“normal”让浏览器选一个合适的line-height，它一般基于字体的大小。

添加上边界，让页脚周围有些空间。

在做练习的时候，检查整个“lounge.css”文件。看一下有没有什么东西要缩写？如果有，继续修改。



休闲室的DJ。



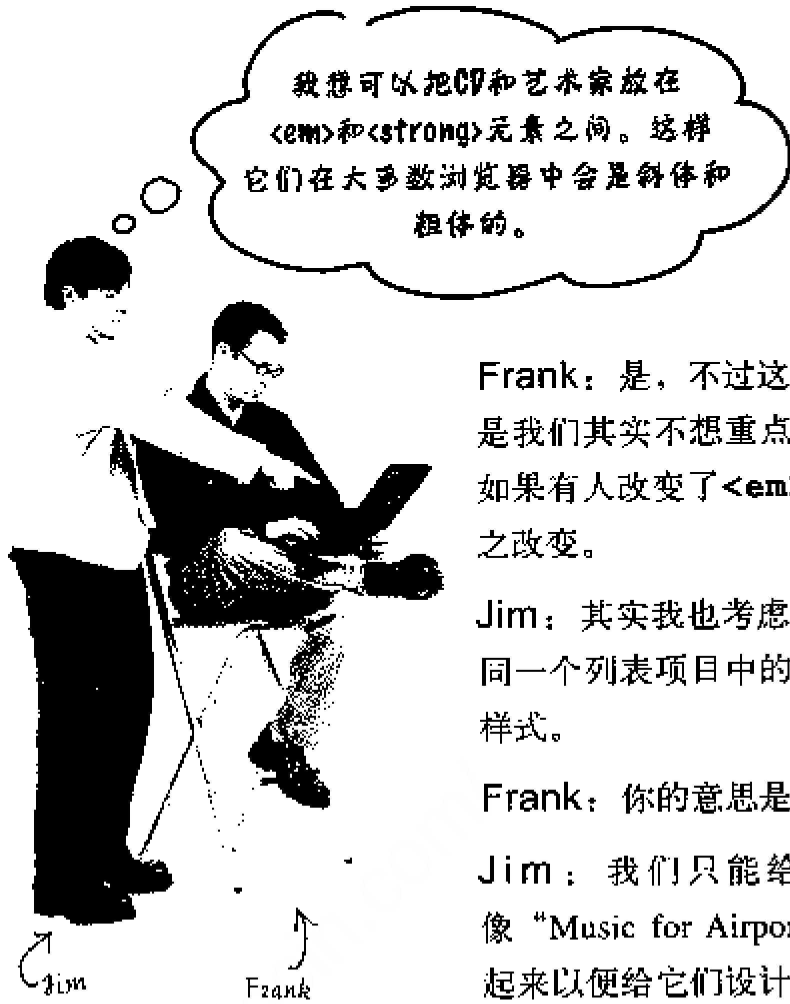
What's playing at the Lounge

We're frequently asked about the music we play at the lounge, and no wonder, it's great stuff. Just for you, we keep a list here on the site, updated weekly. Enjoy.

- *Buddha Bar*, **Claude Challe**
- *When It Falls*, **Zero 7**
- *Earth 7*, **L.T.J. Bukem**
- *Le Roi Est Mort, Vive Le Roi!*, **Enigma**
- *Music for Airports*, **Brian Eno**

BRAIN POWER

你觉得在“休闲室播放的音乐”部分，样式化CD和艺术家的最好方法是什么？



Frank：是，不过这有点像用**<blockquote>**缩排文本。我的意思是我们其实不想重点强调CD和艺术家，只想用斜体和粗体。另外，如果有人改变了**<em>**和**<strong>**的样式呢？CD和艺术家也会随之改变。

Jim：其实我也考虑过这点，但想不到别的方法来做。我是说这是同一个列表项目中的文本，好像没有什么方式可给它们设计不同的样式。

Frank：你的意思是？

Jim：我们只能给元素设计样式，并且现在只有一些文本，像“Music for Airports,Brian Eno”。我们需要一个元素把文本包围起来以便给它们设计不同的样式。

Frank：对，对，我明白你的意思了。

Jim：我想我们可以用一些类似

```
<div class="cd">Music for Airports</div>
<div class="artist">Brian Eno</div>
```

的东西。但这是块元素，会引起换行（linebreak）。

Frank：哦，我觉得你说到点子上了，Jim。还有一个跟**<div>**一样的元素是用于内联元素的，叫做**<span>**，用它可以完美地解决这个问题。

Jim：好极了。它是怎么工作的？

Frank：用**<span>**可以创建一组内联字符和元素。下面，我们来试一试……

用简单的三步添加<span>

<span>元素跟**<div>**工作方式一样，用来把内联内容分成不同的逻辑部分，而**<div>**则是把块级的内容分成不同的逻辑部分。为了弄懂它的用法，我们使用**<span>**来给音乐推荐部分设计样式。先添加**<span>**元素把CD和艺术家包围起来，再写两条CSS规则给**<span>**设计样式。以下是你要做的：

1. 用**<span>**开始和结束标记把CD和艺术家包围起来。
2. 给“cd”类和“artist”类各添加一个**<span>**。
3. 给“cd”类创建一条规则，把它的样式设计成斜体，再给“artist”类创建一条规则，把它的字体设置成粗体。

第一步和第二步：添加<span>

打开“lounge.html”文件找到“Who’s playing at the lounge”标题，在底下你会看到无序推荐列表，如下所示：

每个列表项包括一个CD标题，一个逗号和一个艺术家名。

```
<ul>
<li>Buddha Bar, Claude Challe</li>
<li>When It Falls, Zero 7</li>
<li>Earth 7, L.T.J. Bukem</li>
<li>Le Roi Est Mort, Vive Le Roi!, Enigma</li>
<li>Music for Airports, Brian Eno</li>
</ul>
```

给第一行CD和艺术家添加**<span>**：

先添加一个<span>开始标记、class属性及值“cd”。

然后，在CD标题后添加<span>结束标记。

同样，把艺术家放到<span>元素之间，不同的是类属性值变为“artist”。

```
<ul>
<li><span class="cd">Buddha Bar</span>, <span class="artist">Claude Challe</span></li>
<li>When It Falls, Zero 7</li>
<li>Earth 7, L.T.J. Bukem</li>
<li>Le Roi Est Mort, Vive Le Roi!, Enigma</li>
<li>Music for Airports, Brian Eno</li>
</ul>
```

第三步：给<span>设计样式

继续进行之前，保存文件并重新载入浏览器。跟**<div>**一样，**<span>**默认是没有样式的，所以你会发现没有变化。

现在我们来添加一些样式。在“lounge.css”文件末尾添加以下两条规则：

给cd和artist两个新类各添加一条规则。

把CD的字体样式设置成斜体。

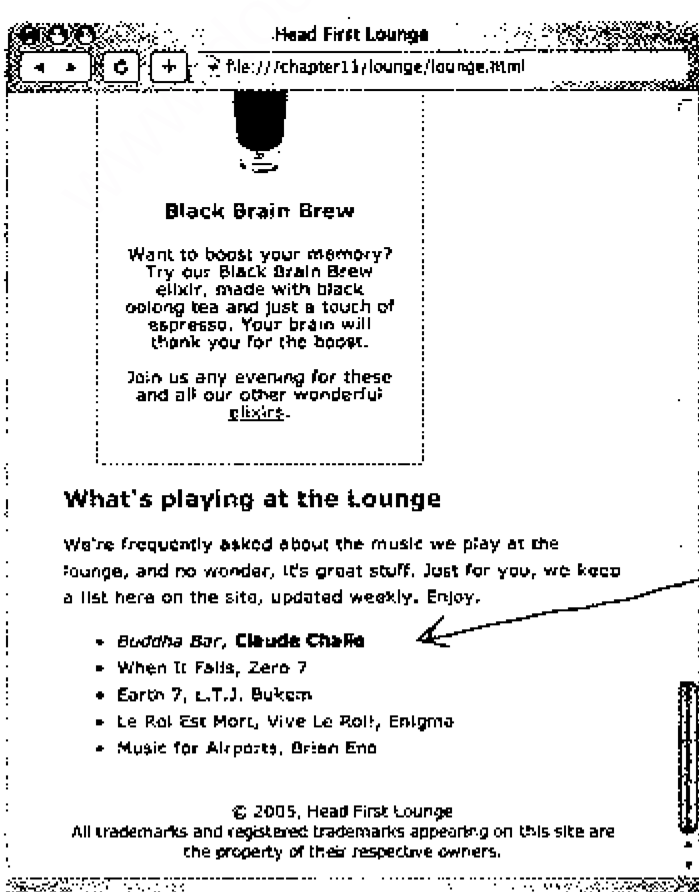
```
.cd {
        font-style: italic;
}

.artist {
        font-weight: bold;
}
```

把艺术家的font-weight设置成粗体。

测试spans

好了，保存并重新载入，结果如下所示：



现在第一个音乐推荐有了正确的样式。



Sharpen your pencil

你的工作是给其余的音乐推荐部分添加<span>元素，并测试页面。答案在本章末尾。

```
<ul>
<li><span class="cd">Buddha Bar</span>, <span class="artist">Claude Challe</span></li>
<li>When It Falls, Zero 7</li>
<li>Earth 7, L.T.J. Bukem</li>
<li>Le Roi Est Mort, Vive Le Roi!, Enigma</li>
<li>Music for Airports, Brian Eno</li>
</ul>
```

there are no Dumb Questions

问： 什么时候该用<span>而不用别的内联元素（如<em>或<strong>）呢？

答： 通常，要用跟内容的意思最匹配的元素来标记内容。所以，如果想强调某些单词，就用<em>；如果想重点强调，就用<strong>。但是，如果你真正的目的只是改变某些文字的样式，比如，一个歌迷网站网页上的专辑或艺术家名，就应该用<span>并且把<span>元素放进适当的类中组成一组，一起设计样式。

问： 可以设置<span>元素的width之类的属性吗？就是说能不能设置一般的内联元素的宽度？

答： 可以设置<span>、<em>、<strong>这些内联元素的宽度，但是只有排版时才能看到效果（下一章将学到如何排版）。也可以给这些元素添加边界和补白，还有边框。内联元素的边界和补白跟块元素的稍有不同——如果在内联元素的四周都添加了边界，只能看到左边和右边的空间。可以在内联元素的上侧和下侧添加补白，不过这些补白不影响周围的其他内联元素的空间，所以补白会跟其他内联元素重叠。

图像跟其他的内联元素稍有不同。width、padding和margin属性更接近于块元素。回忆一下第5章以来讲过的：无论你用<img>元素的width属性还是CSS的width属性设置图像的宽度，浏览器会按比例缩放图像以达到你指定的宽度。这便于你在编辑图像时改变图像在页面上的大小。不过记住，如果你依靠浏览器来缩放图像，就会下载到一些不必要的数据（如果图像比实际需要的大）。

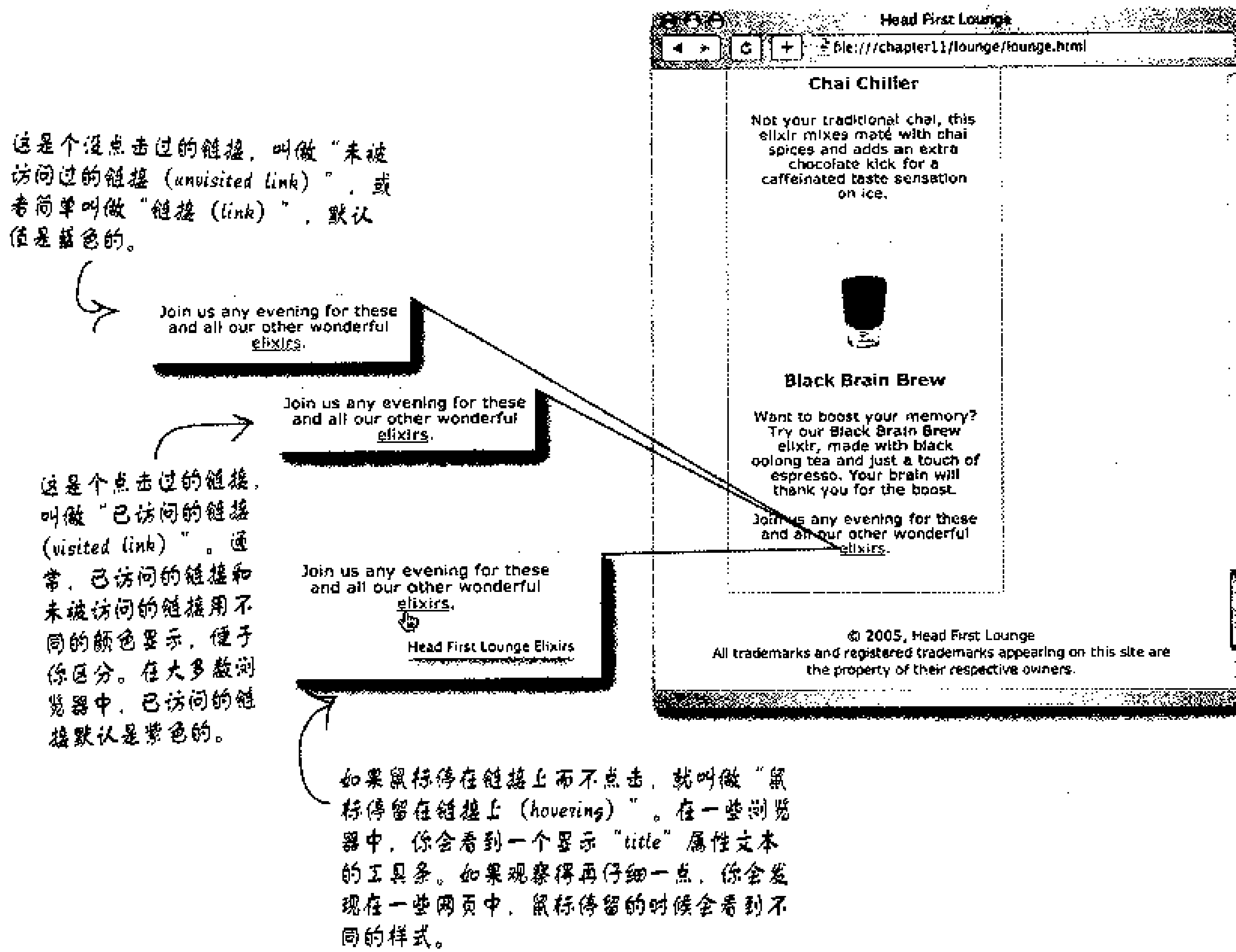


BRAIN POWER

考虑一下<a>元素，它有什么与众不同的样式吗？

<a>元素和它的几个特征

你有没有注意到样式化后链接与之前不同了？链接是元素世界中的变色龙，因为它们会随着情况的不同很快地改变自己的样式。我们仔细看一下：



<a>元素跟别的元素不一样，它的样式随它的状态而改变。如果一个链接从没被点击过，它有一种样式，如果被点击了，就有另外一种样式。如果把鼠标停留在链接上，还会有另一种不同的样式。或许<a>元素还有更多眼睛看不到的样式？我们打个赌……来看一下。

如何根据状态样式化元素？

一个链接可以处于好几种不同的状态：未被访问的、已访问的、鼠标停留的（还有许多别的状态）状态。那么，如何利用这些状态呢？例如，如果能给未被访问的和已访问的状态指定不同的颜色就很不错。或者在用户把鼠标停留在链接上时，链接能够变得更突出。如果有一种方法……

好的，当然有，不过如果我们告诉你它涉及到用伪类(pseudo-classes)，你也许会说今晚已经看得够多了，该休息了，对吗？但是一定要继续！就当我们从来没说过伪类这个词，看看如何样式化链接：

注意元素<a>后面跟着一个“:”，后面是我们想选择的状态。

```
a:link {
    color: green;
}

a:visited {
    color: red;
}

a:hover {
    color: yellow;
}
```

这个选择符选择未被访问的链接。

这个选择符选择已被访问的链接。

这个选择符选择鼠标停留在上面的链接。

练习

把这些规则添加到“lounge.css”文件末尾，保存并重新载入“lounge.html”。试试看链接在每个状态下的显示效果。注意必须清除浏览器的历史记录才能看到未被访问状态的颜色（绿色）。做完之后，在继续操作前务必把这些规则从“lounge.css”文件中删除。

there are no Dumb Questions

问： **如果我只是跟一般元素一样样式化<a>元素会怎样？如：**

```
a{ color:red;}
```

答： 当然可以这么做，不过那样你的链接在各种状态下看起来都是一样的，不过这样用户就不太方便，因为没法区分哪个是已经访问过的，哪个是还没访问的。

问： **还有其他的链接状态吗？**

答： 还有另外两个：focus（聚焦）和active（激活）。focus状态是指浏览器聚焦于链接上的状态。这是什么意思呢？在有些浏览器中可以按Tab键跳过页面上所有的链接。当浏览器停在一个链接上时，这个链接就有了“focus”。active状态发生在当用户第一次点击一个链接时。关于这两个状态的一点警告是：有些浏览器并不能很好地支持它们，所以如果它们对你的设计很重要，那么要对它们进行测试并确定浏览器能支持它们。

问： **链接能同时处于几个状态吗？比如，链接(link)可以是已访问的(visited)，鼠标停留(hover)在上面，并且用户可以马上点击激活(active)它。**

答： 当然能。应用哪个样式取决于规则的顺序。一般认为合理的顺序是：link,visited,focus,hover,active。用这个顺序就能得到你想要的结果。

问： **好的，我明白了。什么是伪类呢？**

答： 它是CSS语言中最易混淆的几个词之一。但是，如你所见，给链接设计样式非常简单。那么，就让我们来介绍一下伪类……



伪类Pseudo-ckass剖析

本周访问：
开始了解伪类。

Head First：欢迎光临，Pseudo-class，很高兴你能来这儿。我必须承认他们第一次让我做这个采访时我没答应。Pseudo-class？令人想到的唯一的东西就是20世纪80年代的Phil Collins歌。

Pseudo-class：哈，那应该是Sussudio。我的名字是Pseudo。

Head First：哎呀！明显的错误。也许我们可以从这方面入手。你能告诉我们Pseudo的来历吗？

Pseudo-class：Pseudo通常指这样一些东西，它们酷似某物，但实际上却有差异。

Head First：姓呢？是类吗？

Pseudo-class：大家都知道CSS类是什么，是一个放元素的组，你可以用它来一起样式化类里的元素。把"pseudo"和"class"放在一起就是Pseudo-class：作用和类一样，但不是真的类。

Head First：如果作用和类一样，为什么又不是真的呢？

Pseudo-class：那好。打开一个XHTML文件，找一下类:visited或:link或:hover。找到后告诉我。

Head First：一个也没找到。

Pseudo-class：:visited，:link和:hover都可以定义样式，就如类一样。所以，它们是伪类。换句话说，可以样式化pseudo-class，但没有人会把它们打进XHTML中。

Head First：那它们怎么工作呢？

Pseudo-class：这得感谢你的浏览器。浏览器仔细检查所有的`<a>`元素，并把它们添加到正确的伪类中。如果一个链接是被访问过的，就把它加进"visited"类中。用户鼠标停留的链接呢？浏览器会把它扔进"hover"类中。哦,没有用户鼠标停留的链接？浏览器就把它从"hover"类中拽出来。

Head First：哇，我从来不知道这些。也就是说所有这些类都在那儿，浏览器从后台添加或删除元素。

Pseudo-class：对，知道这点非常重要。否则你如何给链接设计适合它所处状态的样式呢？

Head First：那么，Pseudo，你是不是只用于链接呢？

Pseudo-class：不，我也用于别的元素。有些浏览器也能支持像激活和停留在其他种类的元素这样的伪类，而且还有其他一些伪类。例如，伪类:first-child是赋值于元素的第一个孩子，像一个`<blockquote>`中的第一段。不过除了:link、:visited和:hover，使用其他伪类时都要注意，因为浏览器不一定都支持它们。

Head First：在这次采访中我的确学到了一些东西。谁知道那首其实叫做"Sussudio"的歌？感谢你来这儿，Pseudo-class！

应用伪类

好了，实话实说。你刚学了这本书中最重要的东西：伪类。为什么这么说？不是因为用它可以样式化一些由浏览器决定属于哪个“类”的元素，比如`:link`或`:first-child`；也不是因为它们给了你基于访问者在用你的页面时发生的事件的有效的元素样式设计方法，比如用:hover；而是因为它有可能让你参加设计会，真正理解并讲解伪类，使你变成类的负责人。我们说的是晋升和奖金……至少会令你的伙伴敬畏。

那么，我们来好好利用一下这些伪类吧。你已经给“lounge.css”添加了一些伪类规则，它们对链接的外观有着神奇的效果。但是它们对休闲室并不是很合适，所以我们稍微修改一下样式：

不错，改变很大，我们把一个子孙选择符和一个伪类结合起来使用。第一个选择符表明要选择所有未被访问的嵌套在id 为“elixirs”的元素里的<a>元素。所以我们只是样式化饮料部分的链接。

```
#elixirs a:link {
    color: #007e7e;
}

#elixirs a:visited {
    color: #333333;
}

#elixirs a:hover {
    background: #f88396;
    color: #0d5353;
}
```

这两个用来设置颜色，未被访问的链接用浅绿色……

……已访问的链接用深灰色。

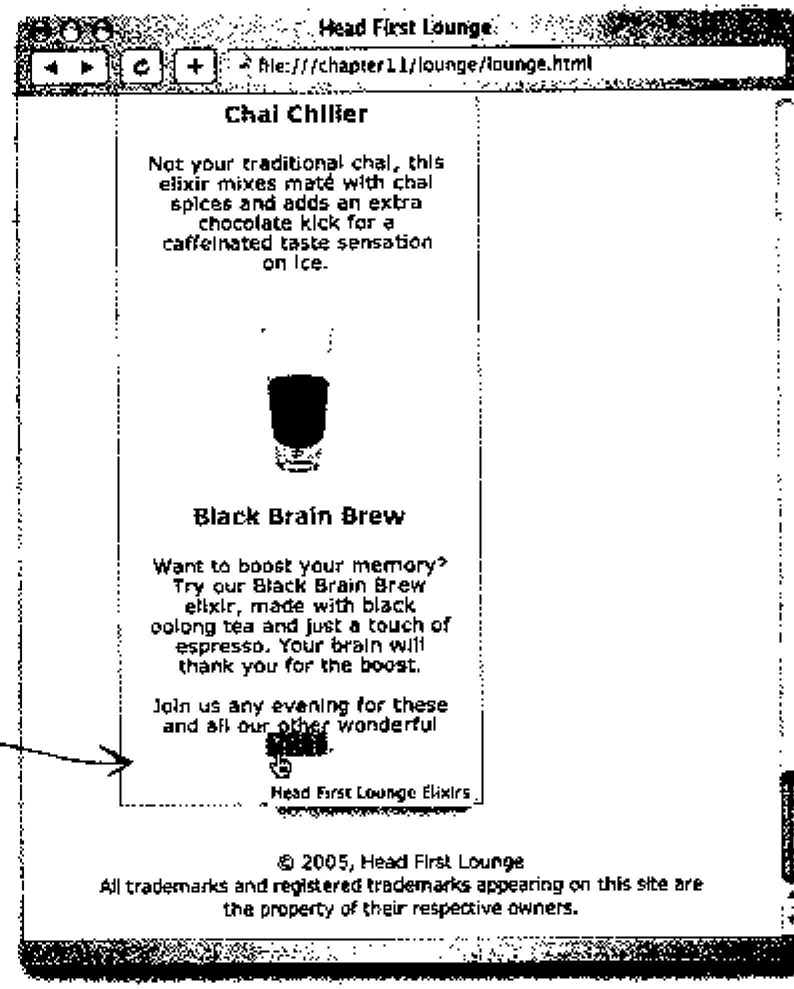
这是相当有趣的规则。当用户把鼠标停留在链接上，背景颜色就会变成红色，这时，链接看起来更突出了。动手试试吧！



打开“lounge.css”，用新的子孙选择符和新的样式重写 a:visited，a:link和a:hover规则。保存，重载并打开页面。

测试链接

重新载入后你会看到饮料部分有一些新样式。谨记，要看未被访问的链接的颜色，必须清除浏览器的历史记录，否则浏览器会保留你对这些链接的访问记录。



现在，未被访问的链接是绿色的，已访问的链接是灰色的，把鼠标停留在链接上时，非常酷的红色使之显得突出。

Sharpen your pencil

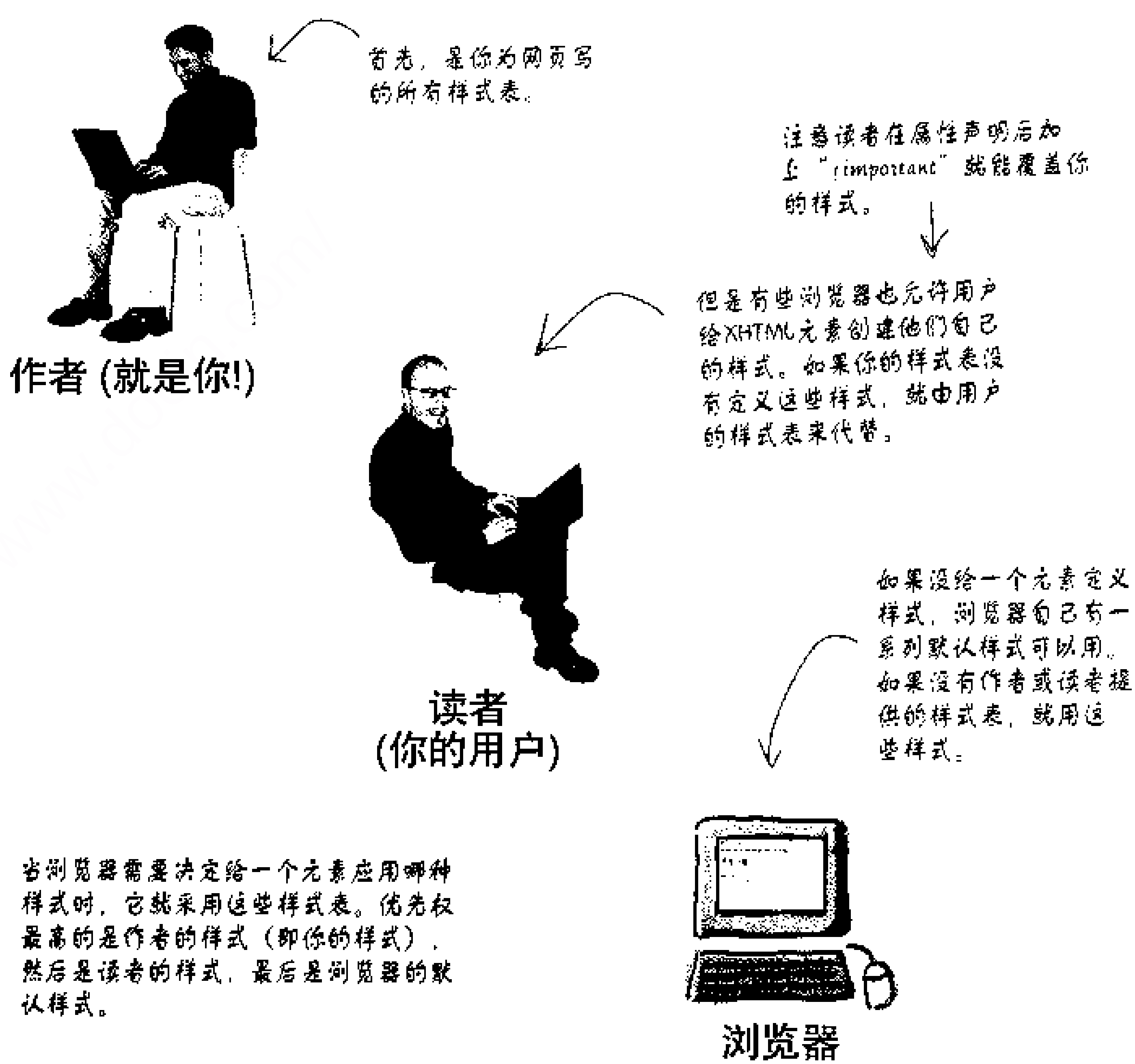
你的工作是给休闲室中的“detailed directions”链接设计样式。跟饮料部分链接一样，我们希望未被访问的链接是浅绿色的，已访问的链接是灰色的。然而，我们不希望休闲室中其他链接有停留状态的样式……跟饮料有些不一样。那么，你该怎么做呢？填空，给“detailed directions”链接设计样式，以后还有一些这种样式的链接要添进休闲室。核对答案（答案在本章末尾），然后动手修改lounge文件。

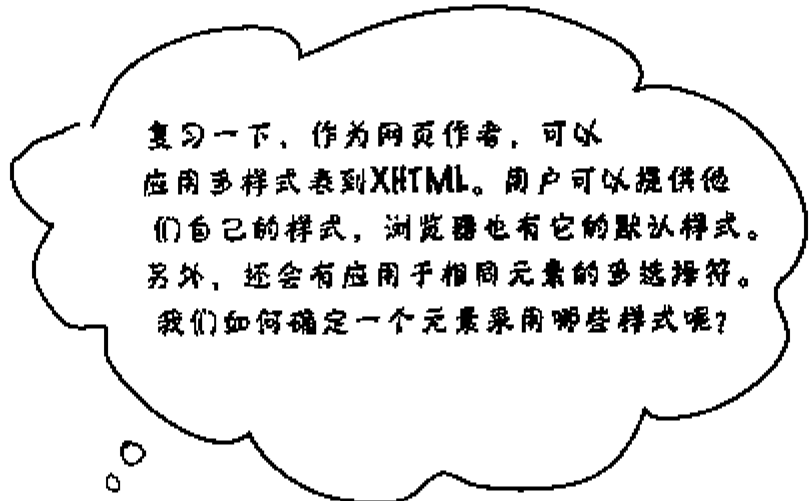
```
______________ { ____________: #007e7e; }
______________ { ____________: #333333; }
```

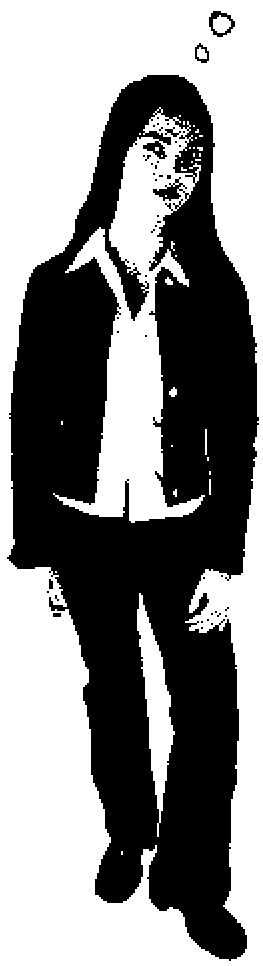
该介绍“层叠（cascade）”了吧？

好的，好的，这本书我们已经学了很多（准确来说是473页），却还没告诉你Cascading Styling Sheet（层叠式样式表）中的“Cascade”是什么。说实话，要完全理解层叠，首先要知道许多关于CSS的东西。不过现在你已经知道不少了，不用再等了。

现在只需要最后一部分信息，你就能理解层叠了。你已经知道用多样式表来更好地组织样式或支持不同类型的设备。但其实当用户访问你的页面时，附近还有另外一些样式表。我们来看一下：





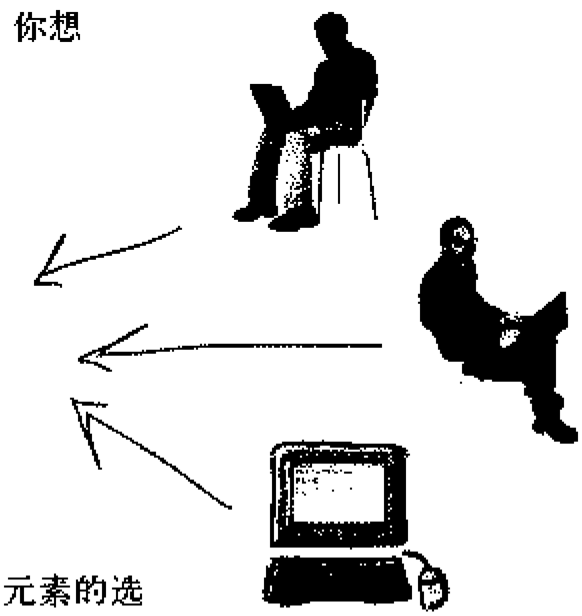


这就是引入层叠的原因。层叠就是浏览器决定采用哪种样式的方法。要回答这个问题，需要把所有东西——附近的所有样式表、规则，以及这些规则中单独的属性声明，集中在一起。

在接下来的两页中我们将分步学习这些工作的细节，包括许多分类，以及如何决定一个元素的哪条规则最具体。你将得到的回报是：仔细研究完后面两页内容，你就能弄清为什么有的样式不以你希望的方式应用，另外，你将比99%的网页制作者懂得更多的层叠（我们不是开玩笑）。

层叠

在这个练习中，你要"扮演浏览器"，假设网页中有一个<h1>元素，你想知道它的**font-size**属性。你可以这么做：



第一步：

把所有的样式表集中起来。

这一步你需要所有的样式表：网页作者写的样式表，读者添加进去的样式表，浏览器默认的样式表（记住，你现在是浏览器，所以你能得到所有这些样式表）。

第二步：

找到所有匹配的声明。

我们要找的是**font-size**属性，所以看看所有的有选择<h1>元素的选择符的**font-size**声明。检查所有的样式表，找出所有匹配<h1>并且有**font-size**属性的规则。

第三步：

将所有匹配结果分类。

所有匹配的规则都找出来后，把它们以作者、读者、浏览器的顺序排序。也就是说，作者写的比读者写的重要，同样，读者的样式比浏览器的默认样式重要。

我们提过，读者可以在CSS属性中加"!important"。如果有这样的属性，排序的时候要把它们放到最前面。

第四步：

根据声明的具体程度排序。

我们在第8章中讲过一些这方面的内容。凭直觉，一条规则越能准确地选择一个元素，就知道它越具体。例如，子孙选择符"blockquote h1"要比"h1"选择符具体，因为它只选择**<blockquote>**中的**<h1>**。下一页我们将学到一个小秘诀，用它可以精确计算一个选择符的具体度。

第五步：

最后，将所有冲突规则按照它们在各自样式表中出现的顺序排序。

列表中的冲突规则排序的依据是，在它们各自样式表中后出现的（更接近末尾）规则更重要一些。也就是说，如果在样式表中添加一条有关某个元素的新规则，它会覆盖前面所有的同属性规则。

就这些了！顺序表中的第一条规则就是获胜者，就用它的**font-size**属性。现在来看如何决定一个选择符的具体度。

欢迎来到“我的具体度是什么”游戏环节

过去用四个数字，但那是在XHTML之前……现在学这个该得你庆幸了。

要计算具体度，先以一组三个数字开始，如下所示：

0 0 0

然后计算选择符中的各种东西，如下所示：

这个选择符有id吗？每个加一分。

这个选择符有类或伪类吗？每个加一分。

这个选择符有元素名称吗？每个加一分。

0 0 0

例如，选择符“h1”中有一个元素，所以结果是：

这个读作数字1。 **0 0 1**

“h1”和“h1.bule”都有一个元素，所以它们的最右边一位数字为1。

再比如，选择符“h1.bule”中有一个元素和一个类，所以结果是：

“h1.bule”还有一个类，所以它的中间一位数字也是1。

这个读作数字11。 **0 1 1**

这两个选择符中都没有id，所以最左边的一位数字都是0。

计算完所有的id、类和元素后，规则的具体度数越大，就越具体。所以，因为“h1.bule”的具体度是11，它比具体度只有1的“h1”更具体。

Sharpen your pencil

小试身手，用以上的规则计算以下选择符的具体度：

h1.greentea	________	ol li p	________	em	________
p img	________	.green	________	span.cd	________
a:link	________	#elixirs h1	________	#sidebar	________

there are no Dumb Questions

问：一个具体度比另一个大是怎么看出来的？

答：把它们当作真正的数字来读就行：100（一百）比010（十）大，010（十）比001（一）大，依次类推。

问：像“h1，h2”这样的规则，应该怎么计算具体度呢？

答：把它想成是两条单独的规则：一条“h1”规则，具体度为“001”，一条“h2”规则，具体度也是“001”。

问：能多讲一些关于“!important”的内容吗？

答：读者可以在属性声明末尾加“!important”忽略一种样式。比如：h1{

fat-size:200%! important;

}

这就会忽略所有同属性的作者样式。

问：得不到读者的样式表，我怎么能领会层叠工作的方法呢？

答：的确不能，不过可以这么看：如果读者忽略了你的样式，其实就是超出了你的控制范围。所以只要用你的样式把页面做成你想要的样子就行了。如果读者选择忽略他们，那么他们将得到他们想要的（要么更好，要么更差）。

融会贯通

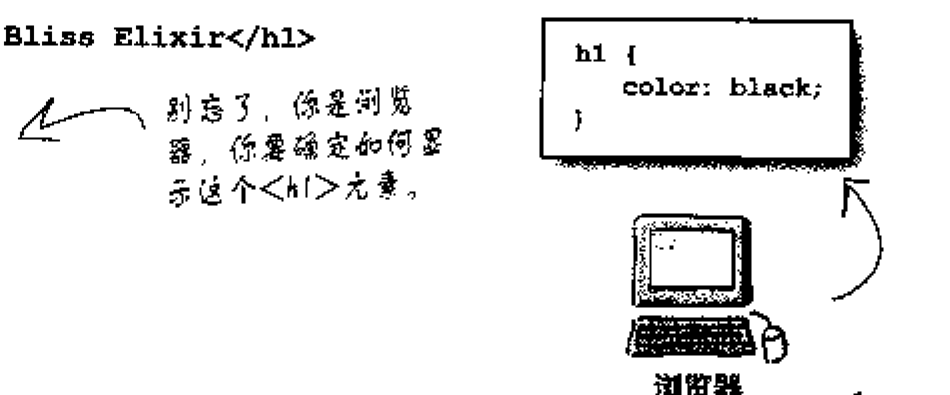
来看一个例子吧。假如你想知道这个**<h1>**元素的**color**属性：

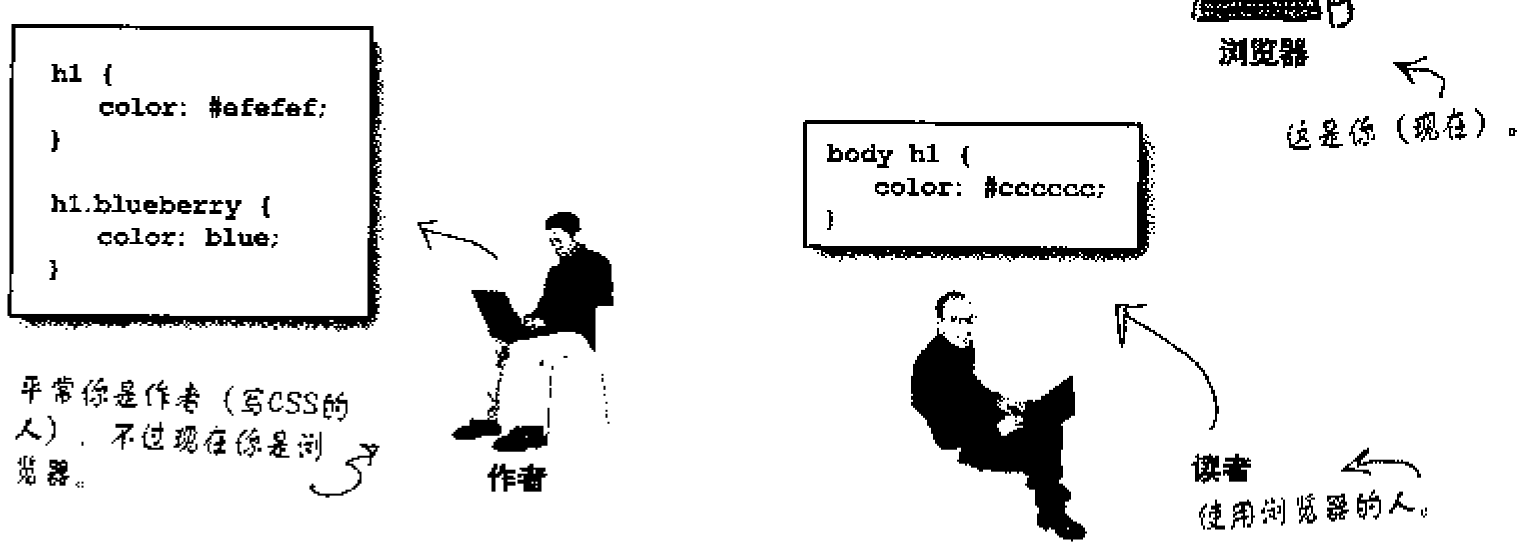
```
<h1 class="blueberry">Blueberry Bliss Elixir</h1>
```

我们用层叠的五个步骤来做：

第一步：

把所有的样式表集中起来。





第二步：

找到所有匹配的声明。

这些是所有可能匹配<h1>元素并且包含color属性的规则。

读者
```
body h1 {
    color: #cccccc;
}
```

浏览器
```
h1 {
    color: black;
}
```

作者
```
h1 {
    color: #efefef;
}

h1.blueberry {
    color: blue;
}
```

第三步：

将所有匹配结果按作者、读者、浏览器的顺序排序。

作者
```
h1 {
    color: #efefef;
}

h1.blueberry {
    color: blue;
}
```

读者
```
body h1 {
    color: #cccccc;
}
```

浏览器
```
h1 {
    color: black;
}
```

以作者、读者、浏览器的顺序重新排序。

第四步：

根据声明的具体程度排序。为此我们需要先算出每个规则的具体度值，然后重新排规则。

```
h1 {
    color: #efefef;
}
```
0 0 1

```
h1.blueberry {
    color: blue;
}
```
0 1 1

```
body h1 {
    color: #cccccc;
}
```
0 0 2

```
h1 {
    color: black;
}
```
0 0 1

blueberry类的规则移到了最前面，因为它的具体度最高。

```
h1.blueberry {
    color: blue;
}
```
0 1 1

```
h1 {
    color: #efefef;
}
```
0 0 1

```
body h1 {
    color: #cccccc;
}
```
0 0 2

```
h1 {
    color: black;
}
```
0 0 1

注意只在作者、读者、浏览器范围内排序，不是重排整个列表，否则“body h1”规则会移到读者设置的“h1”规则之前。

第五步：

最后，将所有冲突规则按照它们在各自样式表中出现的顺序排序。

这样就行了，这个例子中没有冲突的规则。Blueberry的具体值为11，很明显是优胜者。如果有两条规则都得011，较后出现的将是优胜者。

```
h1.blueberry {
    color: blue;
}

h1 {
    color: #efefef;
}

body h1 {
    color: #cccccc;
}

h1 {
    color: black;
}
```

作者

读者

浏览器

有了一个优胜者……

经过选择元素、排序、再排序，判断具体度等艰苦的工作之后，"h1.buleberry"规则升到了第一位，所以<h1>元素中的颜色属性将是蓝色。

there are no Dumb Questions

问：我明白了。一条规则在CSS文件中越靠后，优先权就越高。但是XHTML中的多个样式表链接的优先权又是怎样的呢？

答：不管在不在同一个CSS文件中，顺序总是从头到尾。就当是把CSS以被链接的顺序都一起插入到文件中。

问：当按具体度排序时，不用重排所有的规则吗？

答：不。每次排序，都是以之前的顺序为基础的。首先，以作者、读者、浏览器的顺序排序，接着，在这些分类中按具体度排序，最后，将具体度相同的所有元素，根据在样式表中出现的顺序重新排序。

问：读者真会自己做样式表吗？

答：一般不会。不过也有一些视觉有障碍的读者需要修改所有的东西。不过，因为每个读者只能控制他们看到的东西，其实不会影响到你的设计。

问：我要记住多少这方面的内容啊？

答：对于如何把所有这些元素装配到一起，你会形成某种直觉，在长期的练习中这种直觉会让你变得比较清楚。偶尔，可能你会看到一个在你脑子中犹豫不决的样式突然出现在页面上，这就是练习的必要性了。你能够完成层叠，在理解它之前就确切地知道页面上将出现什么。

如果经过这么多步骤后，还是没找到包含指定属性值的属性声明规则又该怎么办？



问得好。其实我们在第8章中讲过一点儿这方面的内容。如果你在层叠中的规则中没有找到匹配的属性，就要用继承。但不是所有的属性都能通过继承得到，比如边框属性。对于能继承的属性（比如color,font-family,line-height等），浏览器从父亲开始检查元素的祖先，给属性找一个值，如果能找到，就是我们要的属性值。

明白了。嗨，但如果那个属性不能通过继承得到，或者在祖先规则中也找不到值，那又如何？



那么唯一的办法就是求助于浏览器样式表中的默认值，所有的浏览器对每个元素都有默认样式。

到底为什么这个叫做“cascade”（层叠）呢？

选择“cascade”这个名字缘于样式排列的方法，来自多个样式表的样式可以“层叠”向下排列在页面中，每个元素应用最具体的样式。（如果还是不太明白，别担心。我们也没比你清楚多少。只管把它叫做“CSS”，继续向前就可以了）。

停！做完这个练习后再开始下一章！

BRAIN POWER

这是个特殊的思考题，特殊到只有做完才能学习下一章。你需要做的是：

1. 打开文件“lounge.html”，找到饮料`<div>`。
2. 把整个`<div>`部分移到文件开头，即包含休闲室logo的段落后面。
3. 保存并重新载入页面，有什么变化吗？
4. 打开文件“lounge.css”。
5. 找到“#elixirs”规则。
6. 把以下声明添加到上述规则末尾：

```
float: right;
```

7. 保存文件，把页面重新载入浏览器。

有什么变化吗？你觉得这个声明有什么作用？

要点

- <div>元素用来把相关的元素组成逻辑部分。
- 创建逻辑部分有助于标识页面的主内容区、标题和页脚。
- 可以用<div>元素把需要相同样式的元素组成一组。
- 用嵌套的<div>元素给文件添加更深一层的结构，这样有利于结构清晰和样式设计。不过除非真正需要，否则别轻易添加结构。
- 用<div>元素把几部分内容组成一组，就可以跟其他块元素一样给它设计样式。例如，可以用border属性给一组被<div>包围着的元素添加边框。
- width属性用来设置元素内容区的宽度。
- 一个元素的总宽度等于内容区的宽度加上所有补白、边框和边界的宽度。
- 一旦设置了一个元素的宽度，它就不再随浏览器窗口的宽度变化而伸缩了。
- Text-align是一个用于块元素的属性，可以使块元素中的所有内联内容居中。它可以被任何嵌套的块元素继承。
- 可以用子孙选择符选择嵌套在其他元素中的元素，例如，子孙选择符

```
div  h2 { … }
```

选择所有嵌套在<div>元素中的<h2>（包括孩子，孙子等）。

- 可以用缩写方式定义相关的属性。例如，padding-top,padding-right,padding-bottom,padding-left都是关于padding的属性，可以用一个缩写定义，padding。
- Padding,margin,border,background和font属性都可以用缩写规则指定。
- <span>内联元素跟<div>元素一样：它用来把相关的内联元素和文本组成一组。
- 跟<div>一样，也可以把<span>元素添加到类中（或者给<span>元素设置唯一的id）来给它们设计样式。
- <a>元素是有不同状态的元素的一个例子。<a>元素主要的状态有unvisited,visited和hover。
- 可以用伪类单独地给每个状态设计样式。伪类和<a>元素一起最常用的是：:link，用于未被访问的链接，:visited，用于已经访问的链接，:hover,用于停留状态。
- 其他元素支持:hover伪类，一些浏览器也支持:first-child,:active，:focus伪类用于其他元素。



XHTML填字游戏在度假中

因为你有一个思考题要做，我们给这章的XHTML填字游戏放了假。别担心，下一章中它会回来。

Sharpen your pencil 答案

以下是一个标出了所有宽度值的盒子，你的工作是算出整个盒子的宽度。答案如下。

30 + 2 + 5 + 200 + 10 + 2 + 20 = 269

Sharpen your pencil 答案

现在你已经理解宽度了，那么饮料盒子的总宽度是多少呢？首先，内容区是200像素，我们还在左侧和右侧设置了补白，它们也会影响宽度，还有一个设置为“thin”的边框。在大多数浏览器中，thin边框是1像素宽，边界呢？我们只设置了一个左边界值，没有右边界值，所以默认的右边界是0像素。你的工作是计算出饮料<div>的总宽度。答案如下。

20 + 20 + 200 + 1 + 1 + 0 + 20 = 262)

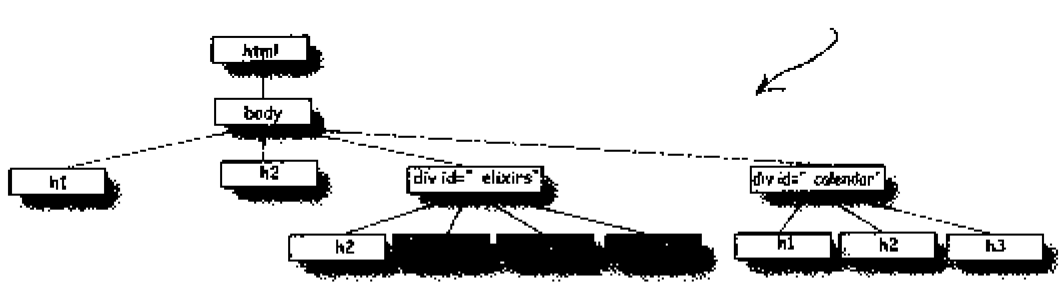
20	20	200	1	1	0	20
左补白	右补白	内容区	左边框	右边框	右边界	左边界

Sharpen your pencil 答案

轮到你了。写出只选择饮料<div>中<h3>元素的选择符。在你的规则中，把颜色属性设置为#d12c47，同时在下图中标出你选的元素。答案如下：

```
#elixirs h3 {
    color: #d12c47;
}
```

这是规则，选择id为elixirs的元素的所有<h3>子孙。



练习答案

该把你的新知识付诸实践了。休闲室的末尾有一小部分版权信息，用它作为页面的页脚。加一个<div>，组成它自己的逻辑部分。完成后用以下属性给它设计样式：

```
font-size: 50%;
text-align: center;
line-height: normal;
margin-top: 30px;
```

文本做成极小的字体。

文本居中。

把line-height设置成“normal”。

添加上边界，让页脚周围有些空间。

用<div>标记包围版权信息。

并给一个名为“footer”的id。

```
<div id="footer">
  <p>
    &copy; 2005, Head First Lounge<br />
    All trademarks and registered trademarks appearing on
    this site are the property of their respective owners.
  </p>
</div>
```

这里是页脚的CSS。

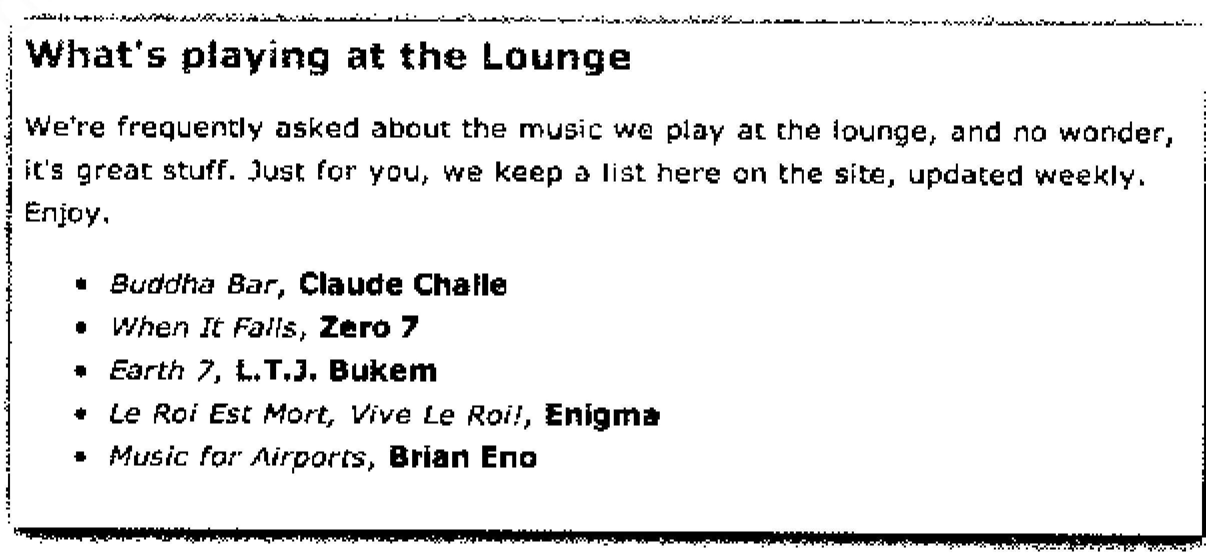
```
#footer {
   font-size: 50%;
   text-align: center;
   line-height: normal;
   margin-top: 30px;
}
```

Sharpen your pencil

答案

你的工作是给其余的音乐推荐部分添加<span>元素，并测试页面。答案如下：

```
<ul>
<li><span class="cd">Buddha Bar</span>,
    <span class="artist">Claude Challe</span></li>
<li><span class="cd">When It Falls</span>,
    <span class="artist">Zero 7</span></li>
<li><span class="cd">Earth 7</span>,
    <span class="artist">L.T.J. Bukem</span></li>
<li><span class="cd">Le Roi Est Mort, Vive Le Roi!</span>,
    <span class="artist">Enigma</span></li>
<li><span class="cd">Music for Airports</span>,
   <span class="artist">Brian Eno</span></li>
</ul>
```



Sharpen your pencil

答案

你的工作是给休闲室中的“detailed directions”链接设计样式。跟饮料部分链接一样，我们希望未被访问的链接是浅绿色的，已访问的链接是灰色的。然而，跟饮料有些不一样，我们希望休闲室中其他元素不要有停留样式……那么，你该怎么做呢？填空，给“detailed directions”链接设计样式，以后还有一些这种样式的链接要添进休闲室。答案如下：

```
a:link      {  color  : #007e7e;  }
a:visited   {  color  : #333333;  }
```

Sharpen your pencil

答案

小试身手，用以上的规则计算以下选择符的具体度。

h1.greentea	0 1 1	ol li p	0 0 3	em	0 0 1
p img	0 0 2	.green	0 1 0	span.cd	0 1 1
a:link	0 1 1	#elixirs h1	1 0 1	#sidebar	1 0 0

12 布局和排版

布置元素



再来教你们一些关于XHTML元素的新方法。我们不再只是让XHTML元素呆坐在那儿……是时候让它们起来帮我们创建一些有真正布局的页面了。怎么样？好的，你已经很熟悉<div>和<span>这两个划分结构的元素了，并且完全了解盒模式是如何工作的，对吗？那么，现在就该用所学知识来做一些真正的设计了。现在不仅介绍关于背景和字体颜色的内容，还要介绍如何用多栏布局进行专业设计。这一章中要用到所有学过的东西。

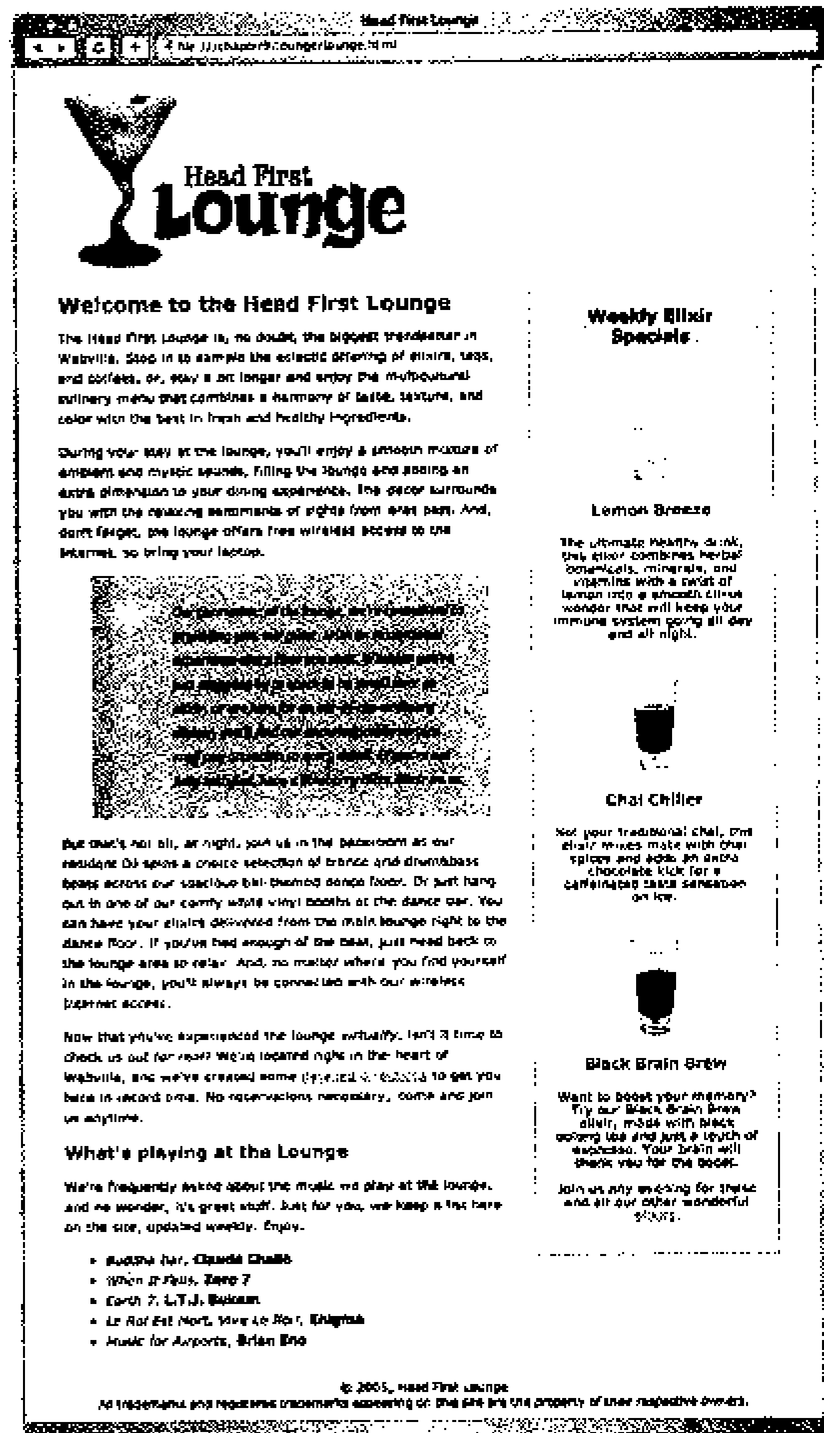
www.docin.com

做思考题了吗？

如果你没做上一章末尾的思考题，那就必须补做了。

回顾上一章的内容，在上一章结束的时候，给你留了一个悬念。我们让你把饮料<div>移到logo下，然后在CSS中给饮料规则部分添加一个如下的属性：

```
float: right;
```



啊，一个属性竟然能有这么神奇的效果！外观普通的页面变成了分成两栏的外观雅致的页面，更具可读性和观赏性了。

神奇的力量是哪儿来的呢？这个看似十分普通的属性是如何造成这么大的效果的呢？能用这个属性让我们的页面变得更有趣吗？当然可以，毕竟这是Head First。不过首先你需要学习浏览器是如何在页面上布置元素的，以及你如何开始在页面上放置元素。

你有一个优势：你已经十分熟悉块元素和内联元素了，甚至也了解盒模式。这些是浏览器如何安排页面的基础。现在你只需知道浏览器如何把所有元素包含在一个页面中，并决定它们放在哪儿。

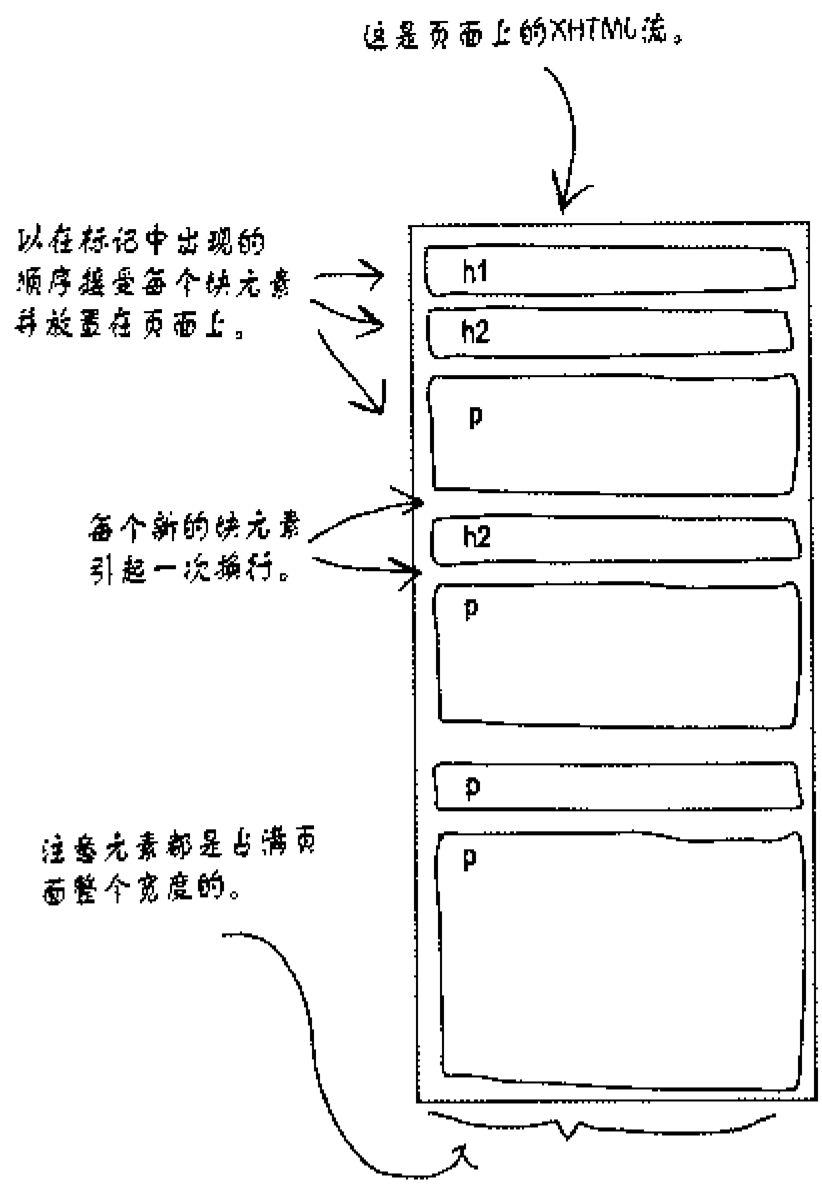
Luke，使用流

流赋予了CSS主人力量。它是神奇的精神力量；它包围着我们；穿透了我们；它把群星聚集在一起……哦，不好意思……

浏览器用流来布置页面上的XHTML元素。浏览器从XHTML文件的开头开始，从头到尾跟着元素的流显示它遇到的每个元素。比如块元素，每两个之间都有换行。所以先显示文件中的第一个元素，接着显示换行，再显示第二个元素，再换行等，从文件的开始一直到末尾。这就是流。

以下是个"缩写的"XHTML。

```
<html>
    <head>...</head>
    <body>
        <h1>...</h1>
        <h2>...</h2>
        <p>...</p>
        <h2>...</h2>
        <p>...</p>
        <p>...</p>
        <p>...</p>
    </body>
</html>
```

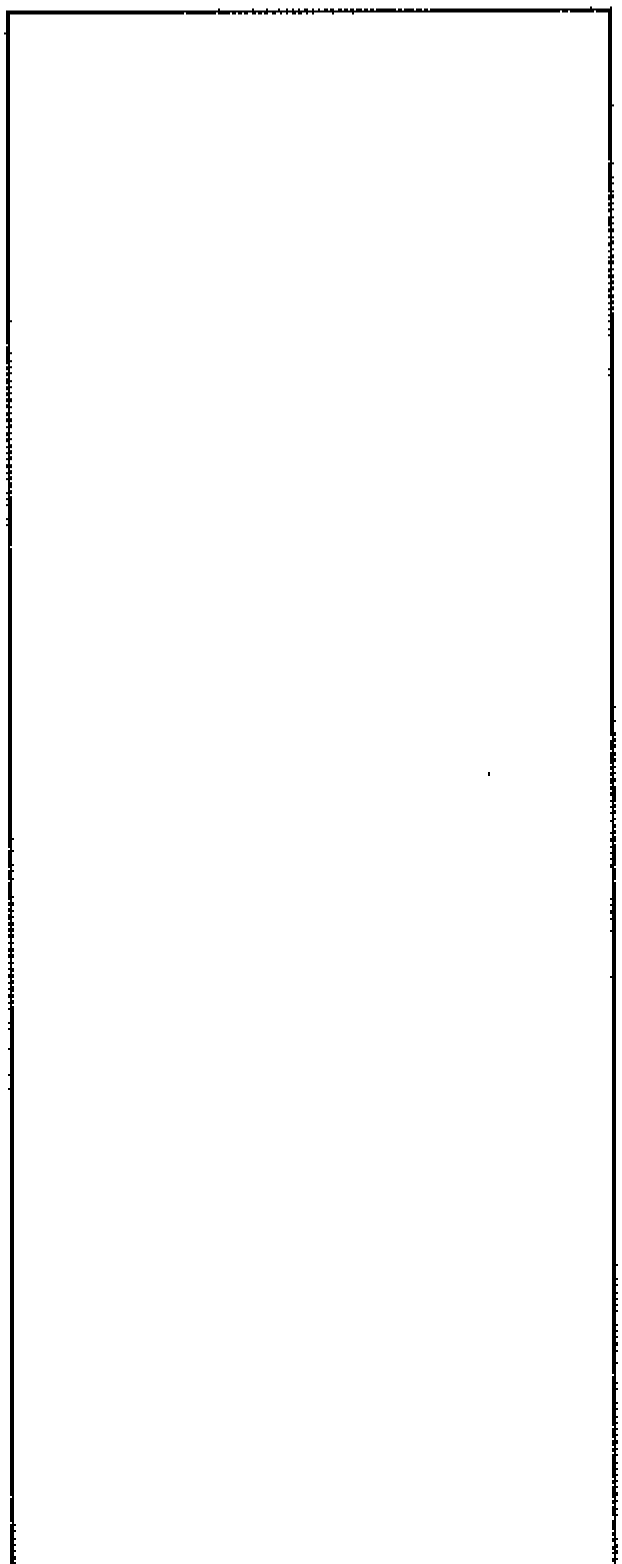


扮演浏览器

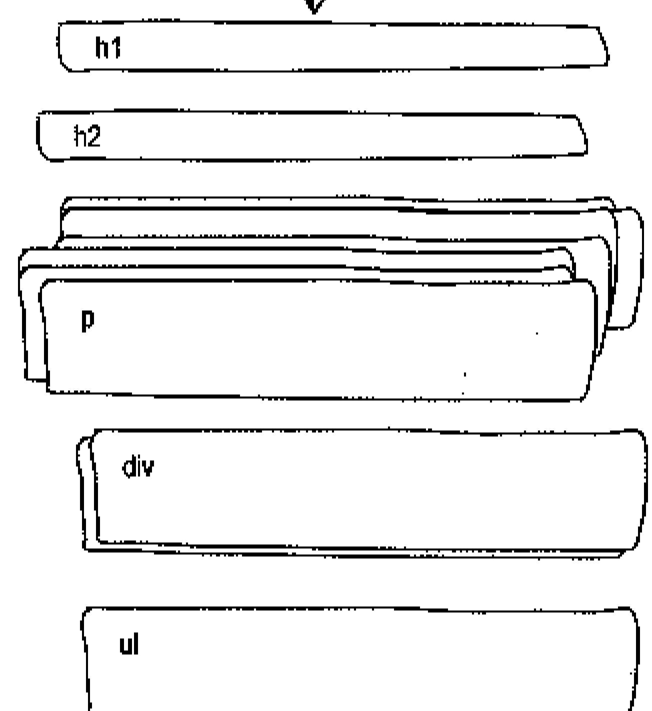


打开"lounge.html"文件并找到所有块元素。把每个元素流入左边的页面中。主要考虑直接嵌套在body元素中的块元素。也可以忽略CSS中的"float"属性，因为你还不知道它是什么。核对答案再前进。

这是你的页面，把"lounge.html"中的块元素"流"到这儿。



以下是你完成工作所需的块元素。



对于内联元素呢？

你知道块元素是从头流到尾的了，每两个元素之间有换行，很简单。那对于内联元素又如何？

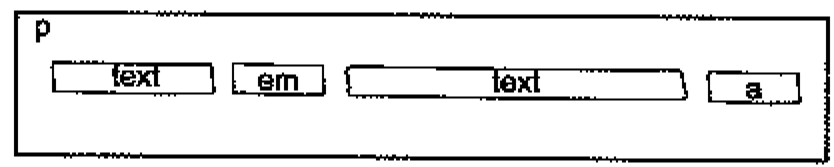
内联元素在水平方向上一个接一个地流，从左上方到右下方。以下是它的工作方法。

这是XHTML的另一小部分。

```
<p>
Join us <em>any evening</em> for
these and all our other wonderful <a
href="beverages/elixir.html" title="Head
First Lounge Elixirs">elixirs</a>.
</p>
```

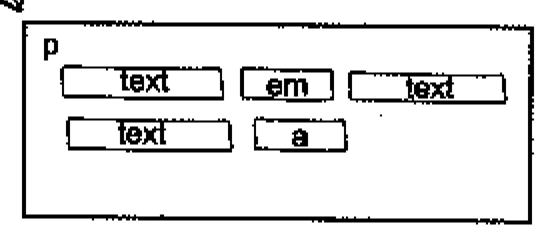
如果我们要让<p>元素的内联内容流到页面上，从左上方开始。

内联元素是在水平方向上一个接一个放置的（只要右边可以有地方放置）。



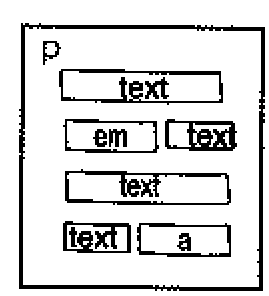
在这里，所有的内联元素都能在水平方向上放置。注意文本是内联元素的一个特殊例子。浏览器把它归为刚好适合于那个空间的内联元素。

如果把浏览器窗口变窄一些，或者用width属性把内容区的大小减小一些又会怎样呢？那么放置内联元素的空间就小了。看看结果如何。



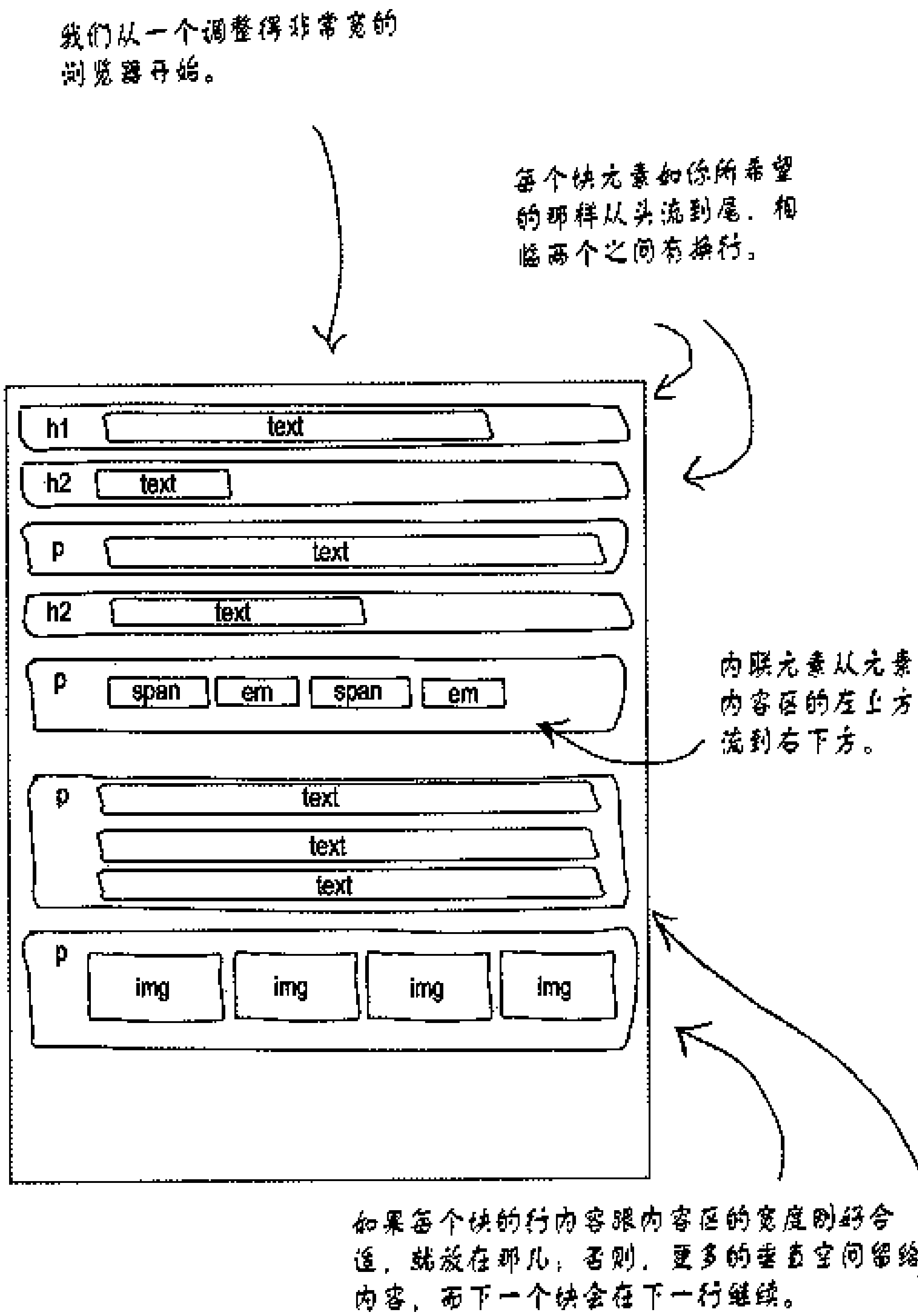
内容从左流到右直到没有足够的空间，就放到下一行。注意浏览器必须把文本分成不同的几部分以放置得更好。

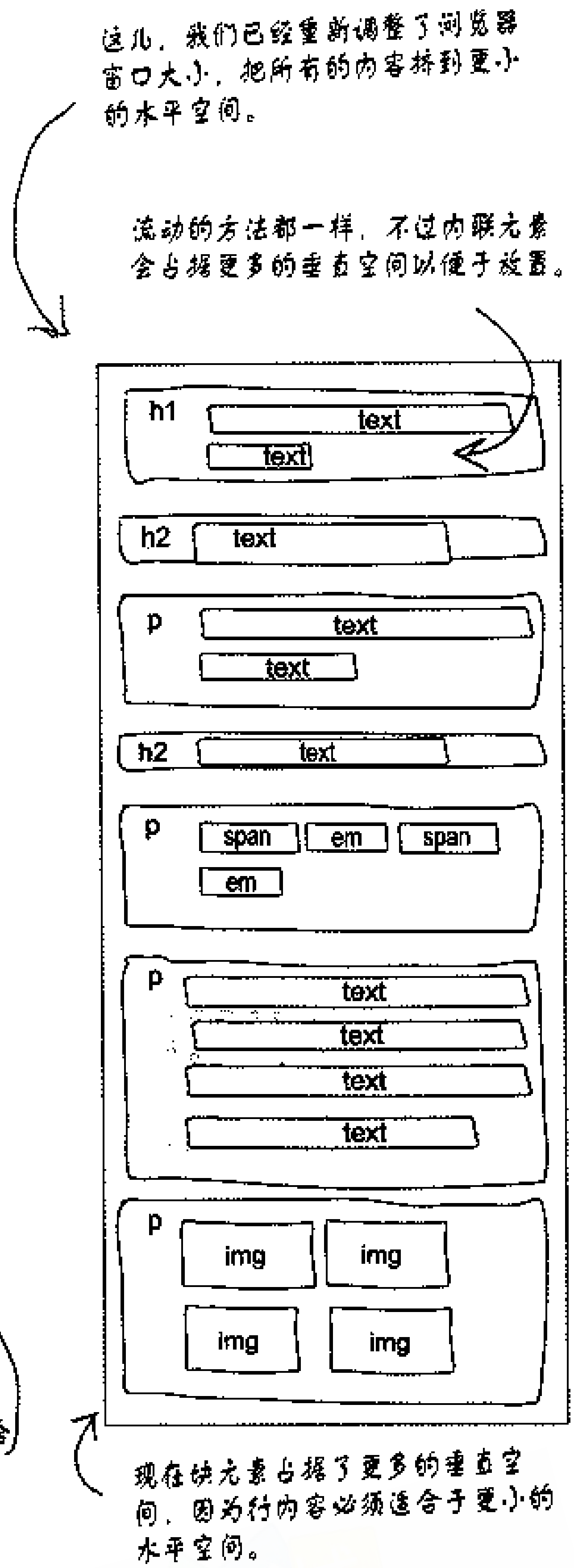
如果我们把内容区变得更窄，看看会发生什么。需要多少行，浏览器就用多少行让内容流入空间。



这些如何一起工作

既然你已经知道块元素和内联元素如何流入，那么让我们把它们放到一起。我们会使用一个典型的有标题、段落和一些内联元素（比如span）、一些强调元素，甚至图像的页面，当然不能忘了内联文本。





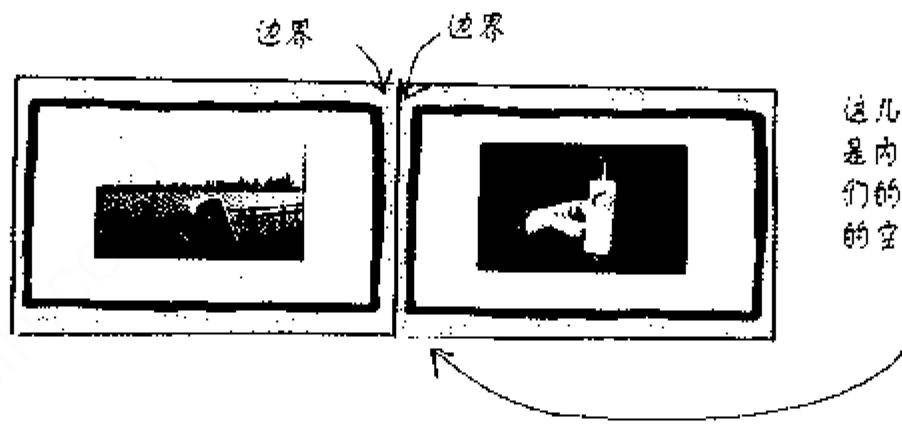
关于流和盒子你还要知道一些东西



我们稍微放大一些来了解浏览器布置块元素和内联元素的另一要点。浏览器根据元素的类型对边界做不同的处理。

当浏览器并排放置两个内联元素时……

当浏览器要并排放置两个内联元素，且这些元素都有边界时，浏览器就会如你所希望的那样做。它在元素之间创建了足够的空间，该空间等于两个边界之和。如果左边的元素有10像素的边界，右边的有20像素的边界，两个元素之间就会有30像素的空间。

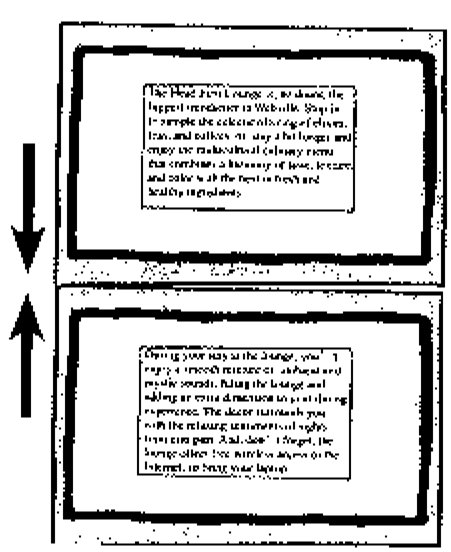


这儿并排放置了两个图像。图像是内联元素，对吗？浏览器用它们的边界之和来计算它们之间的空间。

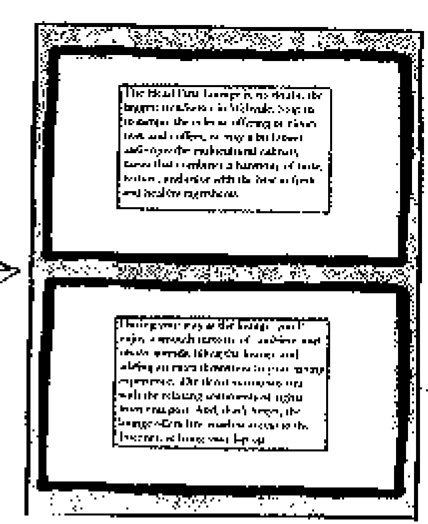
当浏览器并列放置两个块元素时……

这儿事情变得更有趣了。当浏览器并列放置两个块元素时，它把共同的边界重叠到一起。重叠边界的高度是最大边界的值。

当浏览器并列放置两个块元素时，它把边界重叠到一起。



它们共同的边界大小是两个边界中较大的一个。假如上面元素的下边界是10像素，下面元素的上边界是20像素，则重叠的边界就是20像素。



there are no Dumb Questions

问： 如果有一个块元素的边界为0，下面块元素的上边界为20，它们之间的边界就是20吗？

答： 对。如果其中的一个边界较大，就取较大的那个，即使一个边界是0。但如果两个边界相同，比如10像素，它们重叠在一起总共就10像素。

问： 内联元素真能有边界吗？

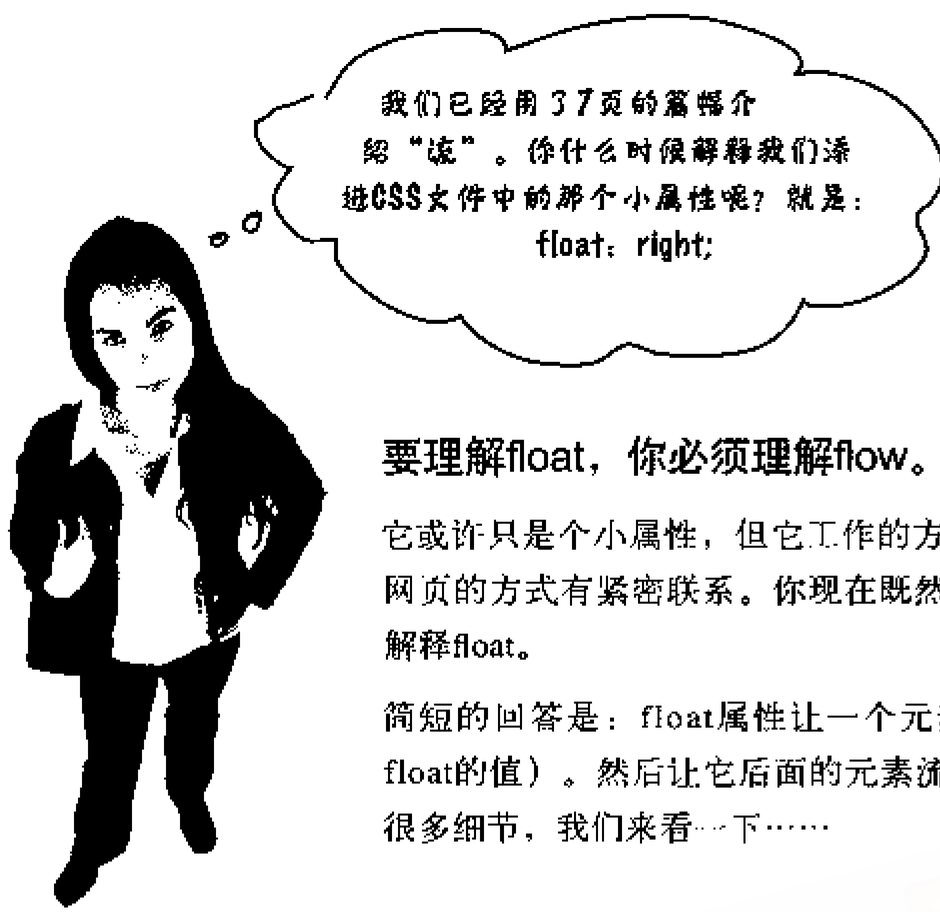
答： 当然能，尽管通常不设置内联元素的边界值。图像就是一个例子，通常不仅设置图像的边界而且设置边框和补白。在这章中我们不会设置任何内联元素的边界，但不久会设置一个边框。

问： 如果一个元素嵌套在另一个元素中，而它们都有边界时会怎样呢？会重叠吗？

答： 是的，会重叠。这种情况下计算边界的方法是：如果两个垂直方向的边界相交，即使一个元素嵌套在另一个元素中，它们也会重叠。注意如果外层的元素有边框，边界就不相交，所以不会重叠，但如果把边框去掉，就会重叠。当你第一次碰到这种情况时会迷惑不解，所以先把这个问题置之脑后吧，等遇到再说。

问： 文本是如何作为内联元素工作的呢？它的内容又不是元素。

答： 即使文本是内容，浏览器也需要让它流到页面上对吧？所以浏览器计算出多少文本适合于给定的行，然后把那行文本看作是内联元素。浏览器甚至可以在它周围创建一个小盒子。如你所见，如果重新调整页面，所有这些块都会随着文本重新装入内容区而改变。



要理解float，你必须理解flow。

它或许只是个小属性，但它工作的方法和浏览器让元素和内容流到网页的方式有紧密联系。你现在既然已经知道这个了，所以我们来解释float。

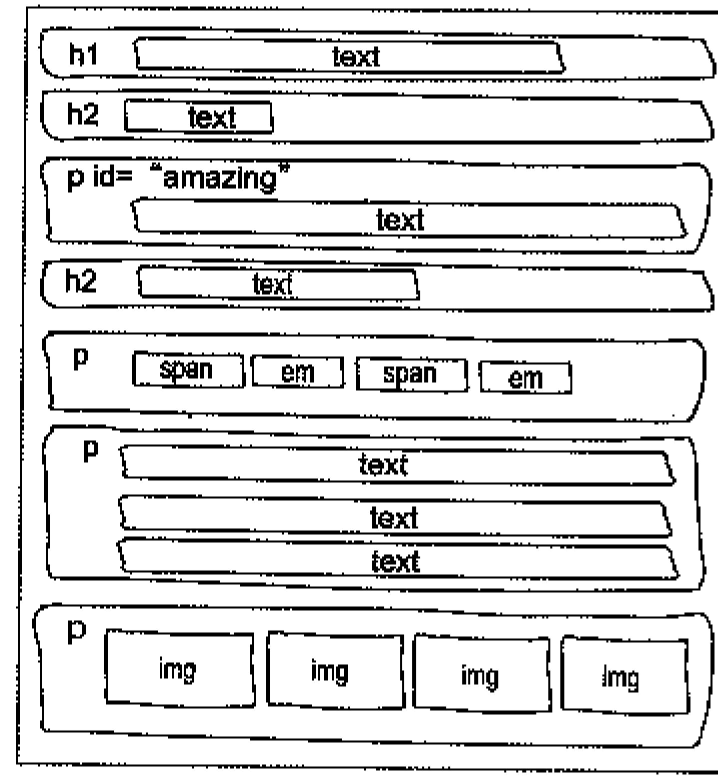
简短的回答是：float属性让一个元素尽可能靠左或靠右（取决于float的值）。然后让它后面的元素流到这个元素的周围。当然还有很多细节，我们来看一下……

如何漂移元素

我们来分步看一个漂移元素的例子，然后看一下它对页面的流做了什么。

首先，给它一个标识符

我们给其中的一个段落一个id。我们想叫它“amazingfloatingparagraph（神奇的漂移段落）”，不过就简称为“amazing”吧。

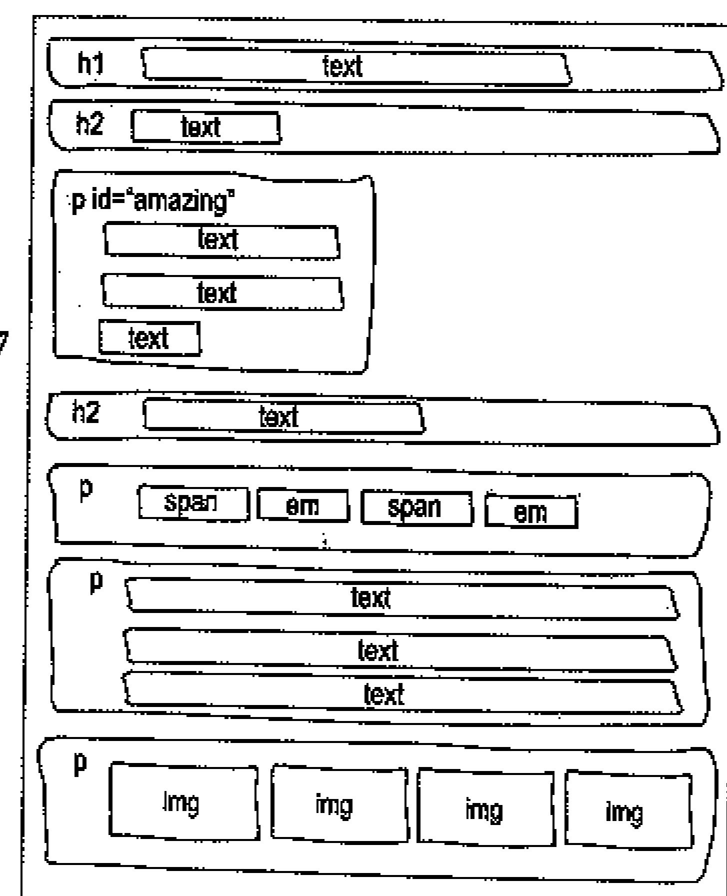


给它设置宽度

漂移元素必须有宽度。我们把这个段落的宽度设置成200像素。规则如下：

```
#amazing {
    width: 200px;
}
```

现在这个段落的宽度是200像素，其中的行内容也被调整到那个宽度。谨记，段落是块元素，所以没有元素会向上移到它旁边，因为所有的块元素前后都有换行。

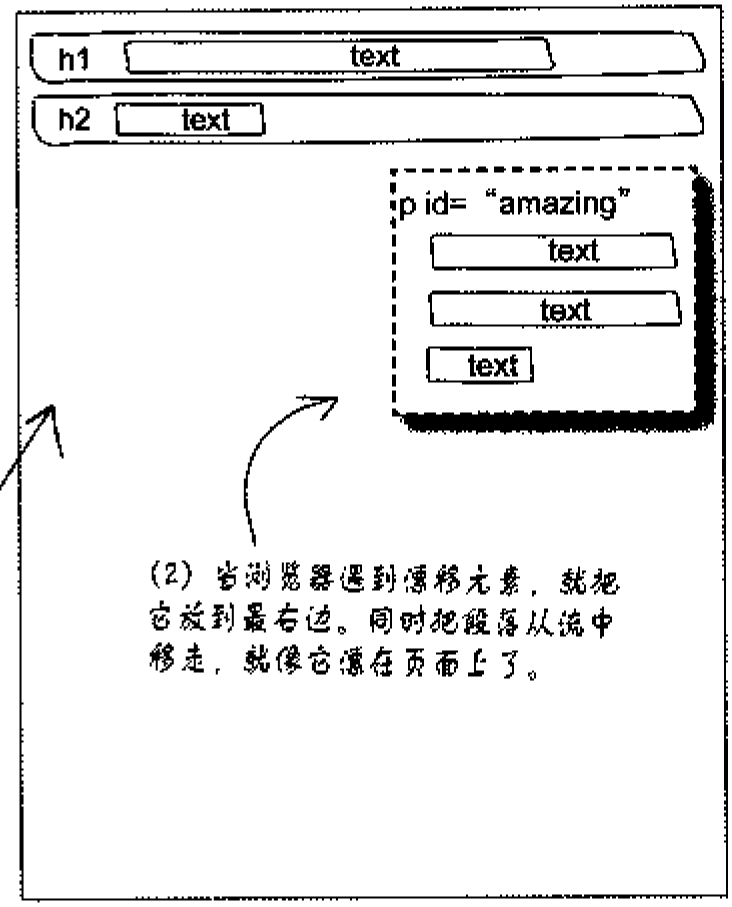


开始漂移

现在我们来添加float属性。float属性值可以设置为left或right。我们还是设置为right：

```
#amazing {
    width: 200px;
    float: right;
}
```

既然我们已经把"amazing"段漂移过去了，我们分步骤看一下浏览器是如何将它和页面上的其他内容流入的。

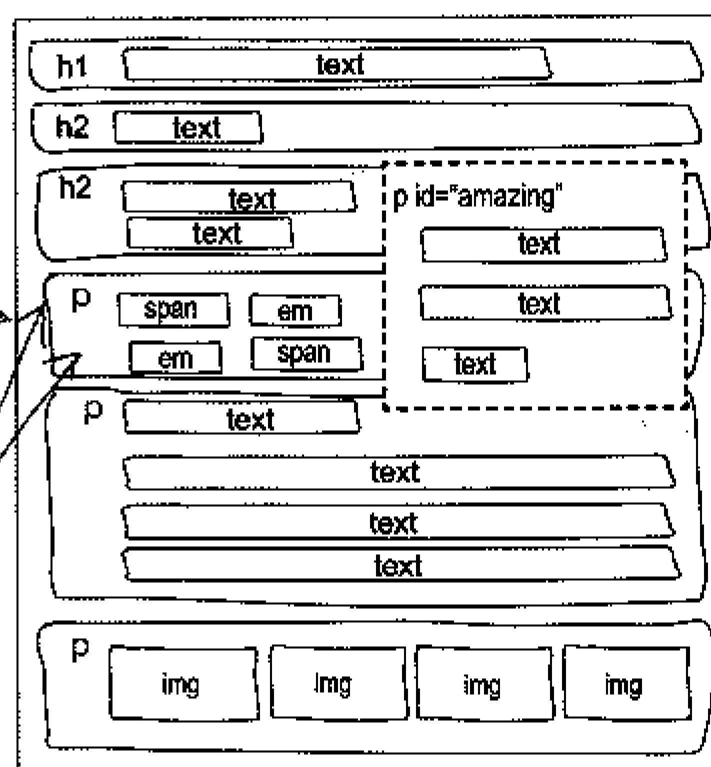


(1) 首先，浏览器跟平常一样往页面流入元素，从文件的开头开始，一直到结尾。

(2) 当浏览器遇到漂移元素，就把它放到最右边。同时把段落从流中移走，就像它漂在页面上了。

(3) 因为把漂移段落从正常的流中移走，并用块元素替代了，那一段落根本就像不在那儿似的。

注意块元素放置在漂移元素之下，因为漂移元素不再是正常流的一部分。



(4) 不过当放置内联元素的时候，不考虑漂移元素的边界，所以它们流到它周围。

然而，当内联元素被注入块元素中时，它们流入漂移元素边界的周围。

在休闲室后台

现在你知道了流及如何在页面上放置漂移元素了。我们再回到休闲室，看看这些都是怎么组合在一起的。

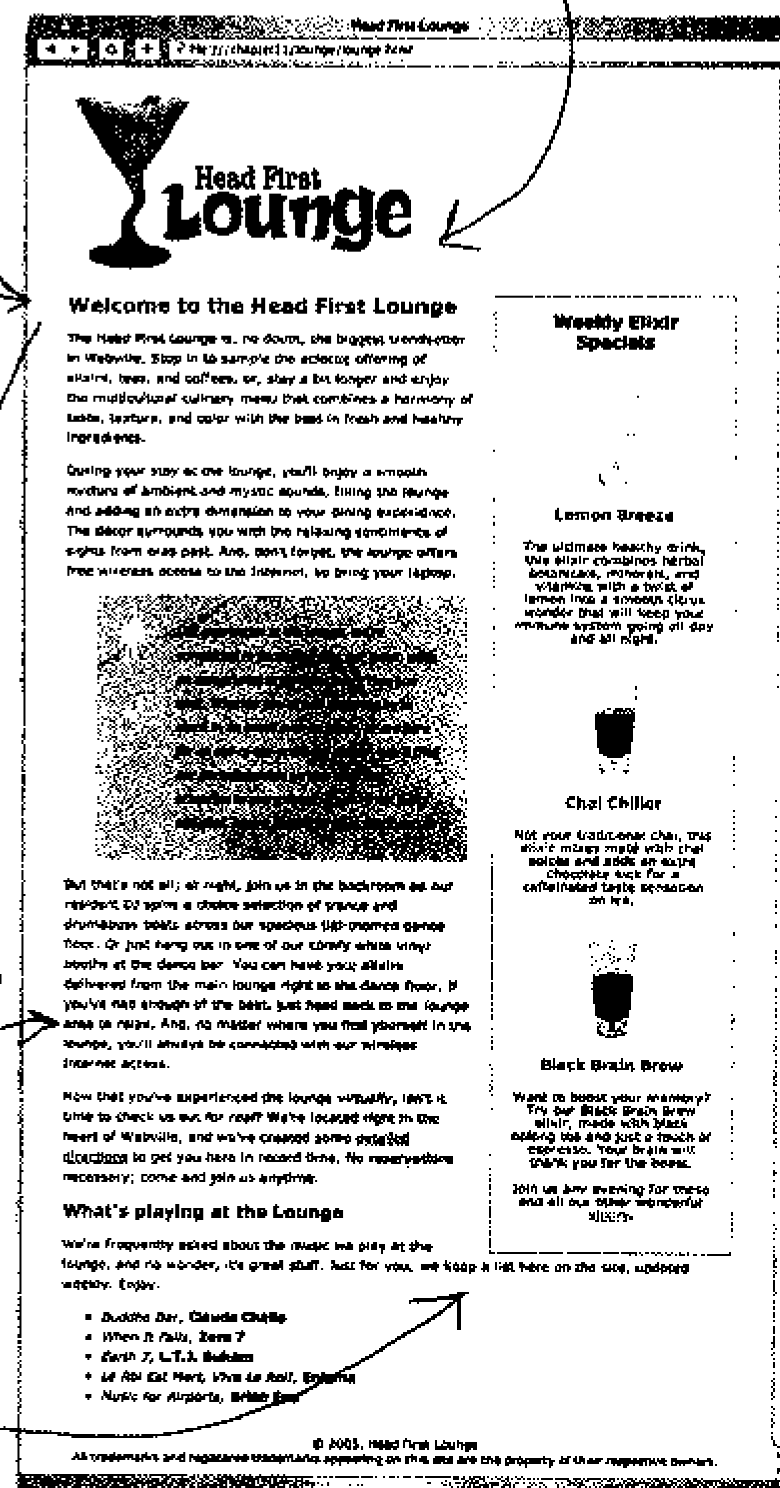
记住，除了把饮料<div>设置为向右漂移外，还把<div>向上移到页面顶部的logo底下了。

把<div>移到最上面就可以让它漂移到右边，然后把整个页面流到它周围。如果我们把饮料<div>留在音乐推荐下面，饮料部分就会在页面的大部分放置完后漂移到右边。

在XHTML中所有这些元素跟在饮料部分后，所以它们流到它周围。

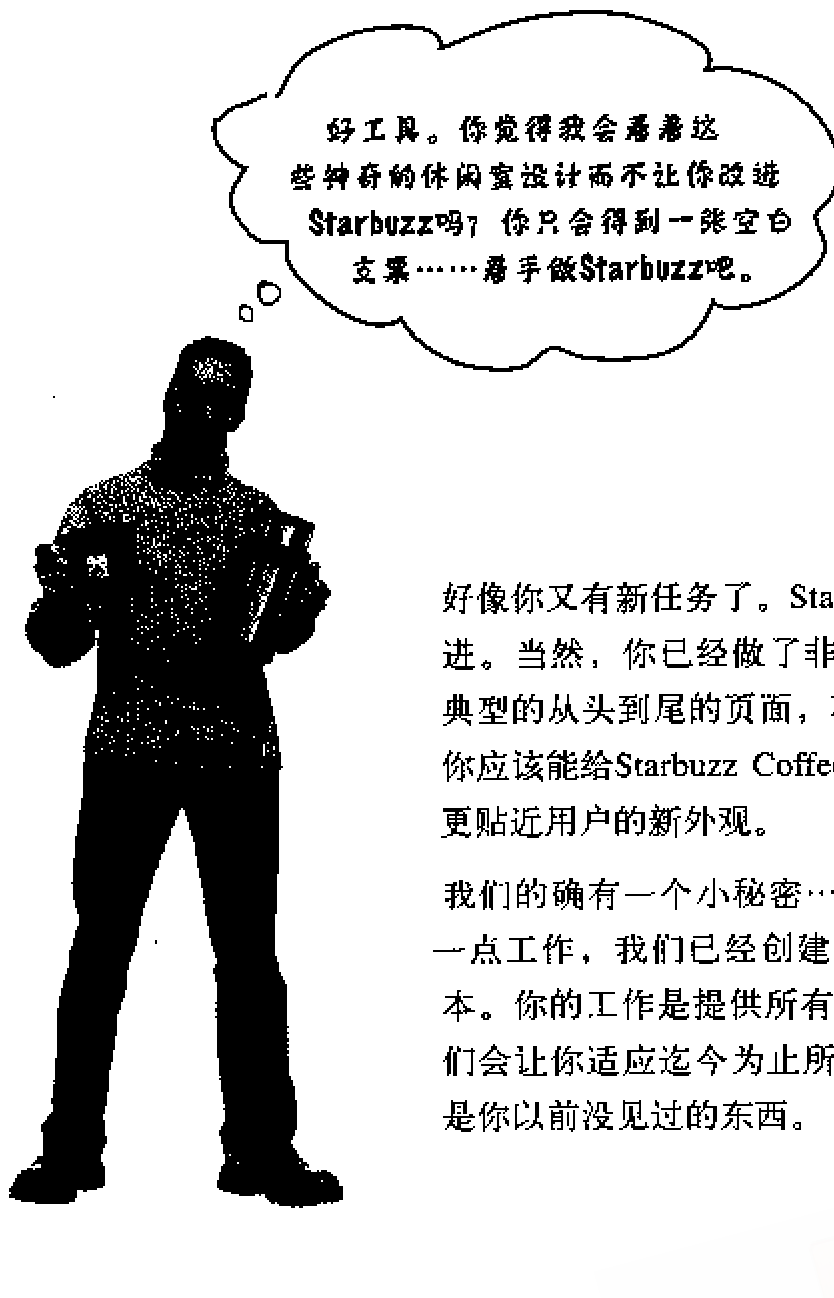
记住饮料<div>被漂移到页面开头。其他所有元素都在它下面，但内联元素在流向页面时不考虑饮料部分的边界。

还要注意文本包围着饮料部分的底部，因为文本被包含在一个跟页面一样宽的块元素中。如果你的饮料部分没被包围，那么把你的浏览器窗口变窄直到文本包围在饮料部分的下面为止。



练习

把饮料<div>移回原来音乐推荐下面的那个位置，保存并重新加载页面。现在那个元素漂到哪儿了？跟后面的答案核对完后把饮料部分移回标题之下。



好像你又有新任务了。Starbuzz确实需要一些改进。当然，你已经做了非常棒的工作，创建了典型的从头到尾的页面，不过既然你知道了流，你应该能给Starbuzz Coffee一个比上一次的设计更贴近用户的新外观。

我们的确有一个小秘密……尽管我们已经做了一点工作，我们已经创建了这个站点的更新版本。你的工作是提供所有的布局。别担心，我们会让你适应迄今为止所做的一切——这些都是你以前没见过的东西。

新的Starbuzz

快速看一下迄今为止我们都做了些什么。从页面外观开始，然后我们浏览一下标记和给它设计样式的CSS。

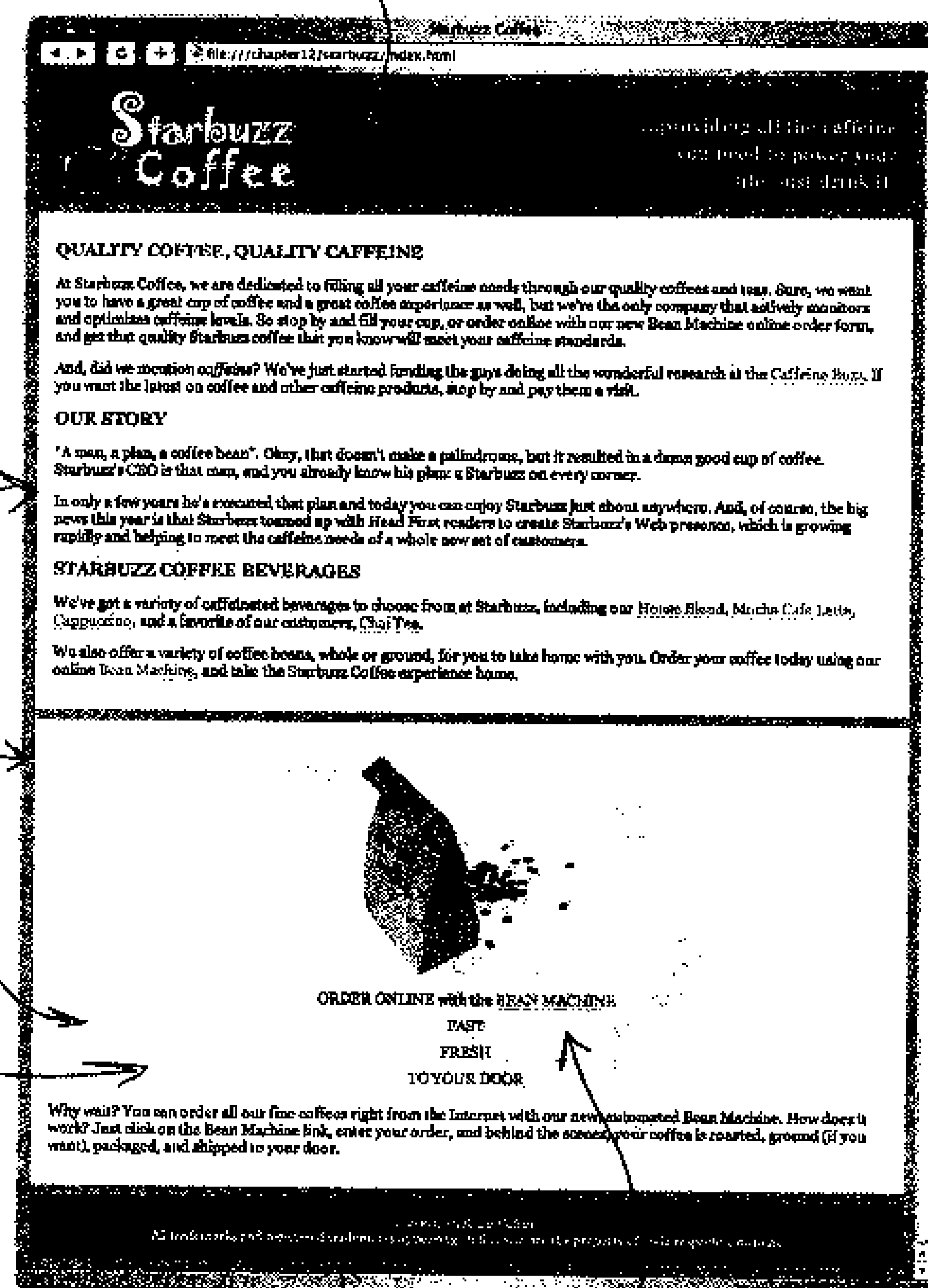
现在有了一个标题，它含有新的漂亮的图片和公司宗旨声明。其实不过是个GIF图像。

我们有四部分：标题、主内容部分、一个叫做“Bean Machine”（磨豆机）的新部分和页脚。

每部分都是可以单独样式化的<div>。

看起来整个页面只用了一种背景颜色，同时每个<div>用一个图像作为背景。

这是“Bean Machine”区。它链接到Starbuzz Coffee的一个新部分，在那儿可以在线订购咖啡豆。这个链接暂时还是无效的，因为你将在以后章节中创建Bean Machine。



这是页脚。它没用背景图像，只用了背景颜色。

注意，我们用一个有趣的方法给链接设计了样式，它有下划虚线……

看一下标记

现在我们来看看新的Starbuzz标记。我们已经把各个逻辑部分放在了一个<div>中，每个都有自己的id。除<div>和<span>之外，剩下的你在第5章中就已经都见过了。所以，迅速浏览一下，熟悉结构，然后翻过本页检查CSS样式。

这是一般的XHTML所必需的。

标题和主内容区后面都跟着<div>。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en" >
<head>
        <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
        <title>Starbuzz Coffee</title>
        <link type="text/css" rel="stylesheet" href="starbuzz.css" />
</head>
<body>

        <div id="header">
                <img src="images/header.gif" alt="Starbuzz Coffee header image" />
        </div>

        <div id="main">
                <h1>QUALITY COFFEE, QUALITY CAFFEINE</h1>
                <p>
                   At Starbuzz Coffee, we are dedicated to filling all your caffeine needs through our
                   quality coffees and teas. Sure, we want you to have a great cup of coffee and a great
                   coffee experience as well, but we're the only company that actively monitors and
                   optimizes caffeine levels. So stop by and fill your cup, or order online with our new Bean
                   Machine online order form, and get that quality Starbuzz coffee that you know will meet
                   your caffeine standards.
                </p>
                <p>
                   And, did we mention <em>caffeine</em>? We've just started funding the guys doing all
                   the wonderful research at the <a href="http://buzz.headfirstlabs.com"
                   title="Read all about caffeine on the Buzz">Caffeine Buzz</a>.
                   If you want the latest on coffee and other caffeine products,
                   stop by and pay them a visit.
                </p>
                <h1>OUR STORY</h1>
                <p>
                    "A man, a plan, a coffee bean". Okay, that doesn't make a palindrome, but it resulted
                    in a damn good cup of coffee.  Starbuzz's CEO is that man, and you already know his
                    plan: a Starbuzz on every corner.
                </p>
                <p>
                    In only a few years he's executed that plan and today
                    you can enjoy Starbuzz just about anywhere. And, of course, the big news this year
                    is that Starbuzz teamed up with Head First readers to create Starbuzz's Web presence,
                    which is growing rapidly and helping to meet the caffeine needs of a whole new set of
                    customers.
                </p>
                <h1>STARBUZZ COFFEE BEVERAGES</h1>
                <p>
                        We've got a variety of caffeinated beverages to choose
```

这儿还有主内容区。

```
                from at Starbuzz, including our
                <a href="beverages.html#house" title="House Blend">House Blend</a>,
                <a href="beverages.html#mocha" title="Mocha Cafe Latte">Mocha Cafe Latte</a>,
                <a href="beverages.html#cappuccino" title="Cappuccino">Cappuccino</a>,
                and a favorite of our customers,
                <a href="beverages.html#chai" title="Chai Tea">Chai Tea</a>.
        </p>
        <p>
                We also offer a variety of coffee beans, whole or ground, for you to
                take home with you.  Order your coffee today using our online
                <a href="form.html" title="The Bean Machine">Bean Machine</a>,
                and take the Starbuzz Coffee experience home.
        </p>

    </div>

    <div id="sidebar">
        <p class="beanheading">
            <img src="images/bag.gif" alt="Bean Machine bag" />
            <br />
            ORDER ONLINE
            with the
            <a href="form.html">BEAN MACHINE</a>
            <br />
            <span class="slogan">
                FAST <br />
                FRESH <br />
                TO YOUR DOOR <br />
            </span>
        </p>
        <p>
          Why wait?  You can order all our fine coffees right from the Internet with our new,
          automated Bean Machine.  How does it work?  Just click on the Bean Machine link,
          enter your order, and behind the scenes, your coffee is roasted, ground
          (if you want), packaged, and shipped to your door.
        </p>
    </div>

    <div id="footer">
        &copy; 2005, Starbuzz Coffee
        <br />
        All trademarks and registered trademarks appearing on
        this site are the property of their respective owners.
    </div>

</body>
</html>
```

这是Bean Machine的<div>。我们给它一个id叫"sidebar"。嘿，想知道是什么意思吗？

最后，页脚也加<div>。

看一下样式

我们看一下用来样式化新的Starbuzz页面的CSS。分步骤来仔细观察CSS。新的Starbuzz看上去有些改进，而你会发现全是些你已经知道的简单的CSS。

```
body {
        background-color: #b5a789;
        font-family:      Georgia, "Times New Roman", Times, serif;
        font-size:        small;
        margin:           0px;
}

#header {
        background-color: #675c47;
        margin:           10px;
        height:           108px;
}

#main {
        background: #efe5d0 url(images/background.gif) top left;
        font-size:  105%;
        padding:    15px;
        margin:     0px 10px 10px 10px;
}

#sidebar {
        background: #efe5d0 url(images/background.gif) bottom right;
        font-size:  105%;
        padding:    15px;
        margin:     0px 10px 10px 10px;
}

#footer {
        background-color: #675c47;
        color:            #efe5d0;
        text-align:       center;
        padding:          15px;
        margin:           10px;
        font-size:        90%;
}

h1 {
        font-size: 120%;
        color:     #954b4b;
}

.slogan { color: #954b4b;}

.beanheading {
        text-align:  center;
        line-height: 1.8em;
}
```

首先我们在body中创建一些基本元素：背景颜色，字体，并且把body的边界设置为0。这确保页面周围没有多余的空间。

接着我们给每个部分创建一条规则。分别给各部分调整字体，添加补白和边界，并且在main和sidebar中指定背景图像。

接着设置标题的字体和颜色。

然后设置叫做slogan类的颜色，把它用在sidebar<div>中。也设置了beanheading类，它也用于sidebar<div>。

```
a:link {
        color:                #b76666;
        text-decoration:      none;
        border-bottom:        thin dotted #b76666;
}
a:visited {
        color:                #675c47;
        text-decoration:      none;
        border-bottom:        thin dotted #675c47;
}
```

在Starbuzz CSS的最后两条规则中我们用一个a:link和一个a:viseted伪类来样式化链接。

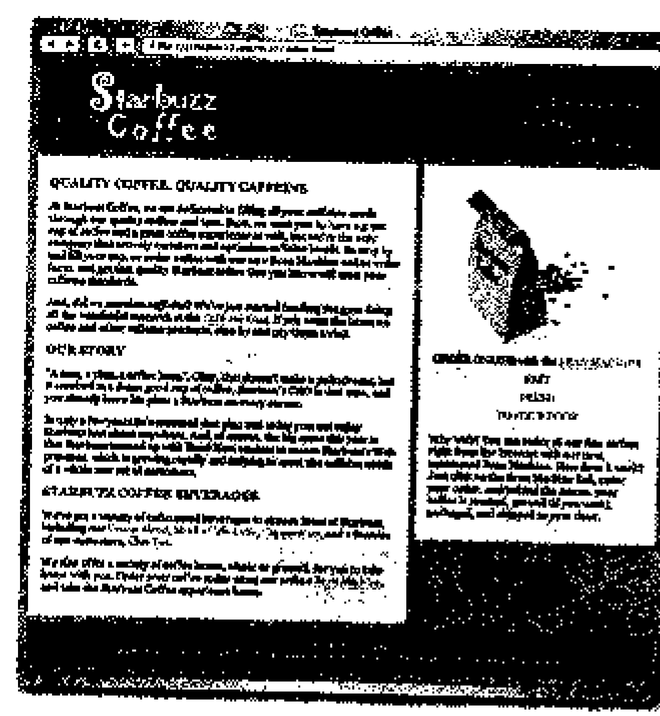
注意我们用dotted下边框代替下划线来让链接有好看的虚下划线效果。这是一个在内联元素里使用边框属性的很好例子。

我们用缩写格式设置border-bottom。

升级Starbuzz

目标是：把Starbuzz Coffee变成如右图所示的站点。为此，我们需要把Bean Machine sidebar移到右边以得到比较美观的分成两栏的页面。好的，你已经在休闲室中做过一次了，对吗？所以，以它为基础，以下是你要做的：

1. 用id给你要漂移的元素一个独特的名字。已经有了。
2. 你想把它漂移到哪个元素下面，务必把它的XHTML移到那个元素后面，在此例中是Starbuzz header。
3. 设置元素的宽度。
4. 把元素漂移到左边或右边。看起来你想移到右边。



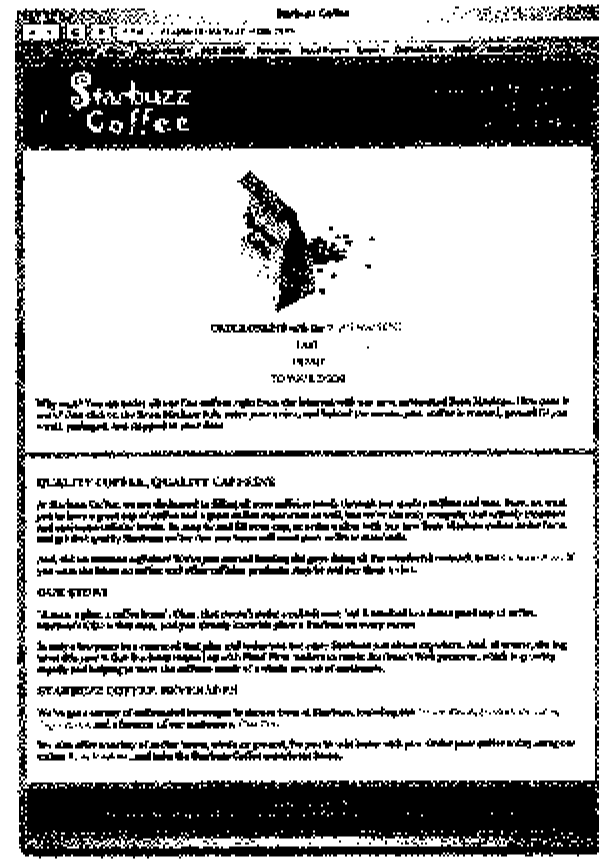
我们用分栏技术得到了分成两栏的好看的外观。

我们开始吧。用这简单的几步，我们就会让Starbuzz CEO感到满意的。

把sidebar移到标题下

有一点是肯定的，如果你把一个元素漂移到另一个元素之后，那么它们的XHTML也要遵循相应的先后顺序。在这个例子中，sidebar要在标题之后。所以，继续，在你的编辑器中找到sidebar`<div>`并把整个`<div>`移到标题`<div>`之后。你会在“chapter12/starbuzz”文件夹中的“index.html”文件中找到XHTML,完成以上操作，保存并重新加载网页。

现在sidebar应该在主内容区的上面。



设置sidebar的宽度并漂移它

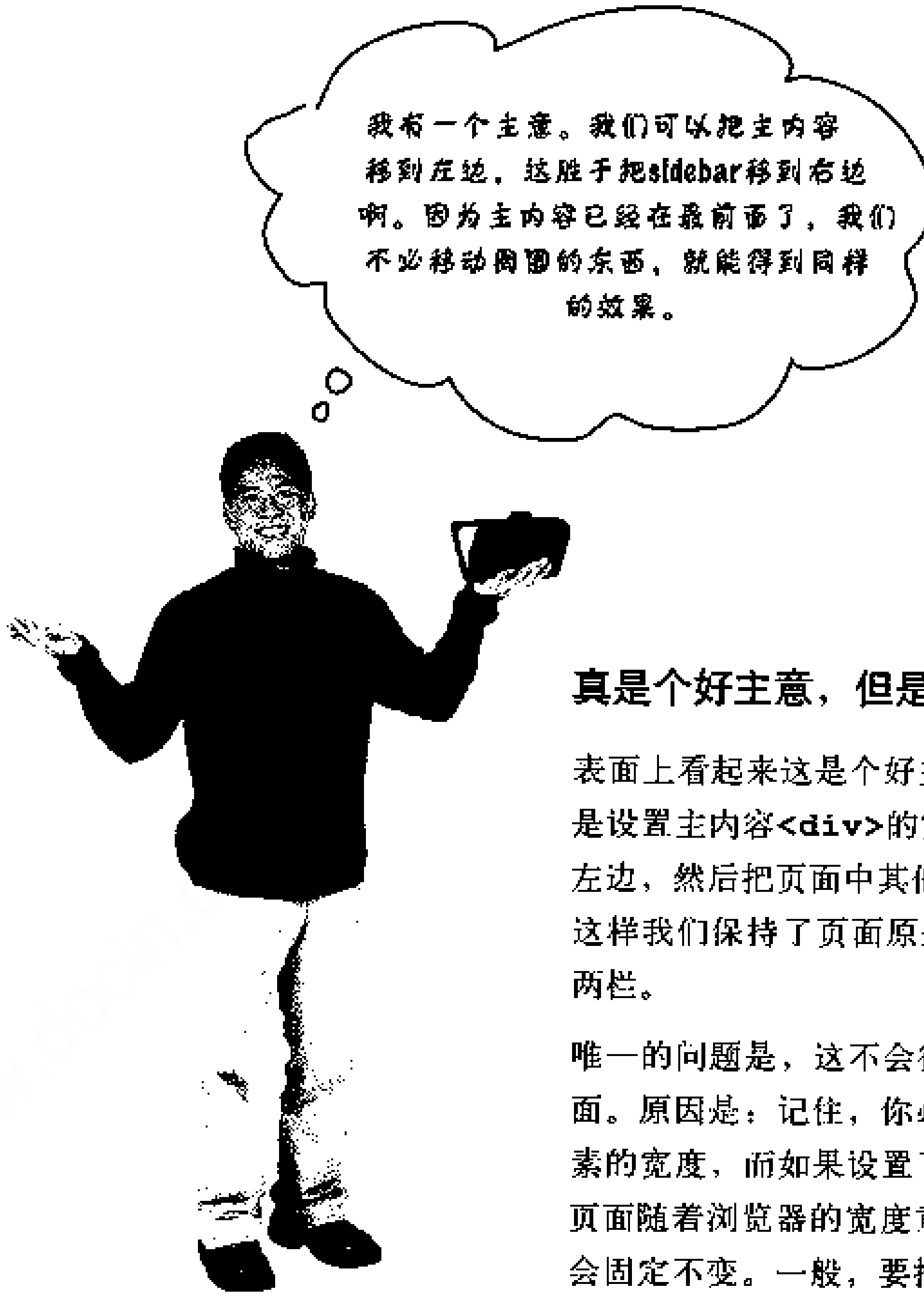
我们把sidebar的宽度设置为280像素。为了漂移sidebar，添加float属性，像这样：

用id选择符选择id为“sidebar”的元素，我们知道这是sidebar的<div>。

```
#sidebar {
        background:  #efe5d0 url(images/background.gif) bottom right;
        font-size:   105%;
        padding:     15px;
        margin:      0px 10px 10px 10px;
        width:       280px;
        float:       right;
}
```

我们把sidebar的宽度设置为280像素。

然后把sidebar漂移到右边。记住，这会把sidebar移到标题之下，尽可能靠右，并且把sidebar从正常的流中移走。XHTML中sidebar后面所有的内容都将上升到它周围。



真是个好主意，但是有一些问题。

表面上看起来这是个好主意。我们要做的只是设置主内容<div>的宽度，并把它漂移到左边，然后把页面中其他的元素流到它周围。这样我们保持了页面原来的顺序，并分成了两栏。

唯一的问题是，这不会得到一个很美观的页面。原因是：记住，你必须设置要漂移的元素的宽度，而如果设置了内容区的宽度，当页面随着浏览器的宽度重新调整时它的宽度会固定不变。一般，要把sidebar设置得比主内容区窄一些，如果它们太宽，看起来就会很糟糕。所以，在大多数设计中，你想让主内容区得到扩展，而不是sidebar。

我们将看到一种采用这种思路，而且效果很好的方法，因此，别抛弃这种想法。我们还要多讲一些考虑各部分放置顺序的原因。

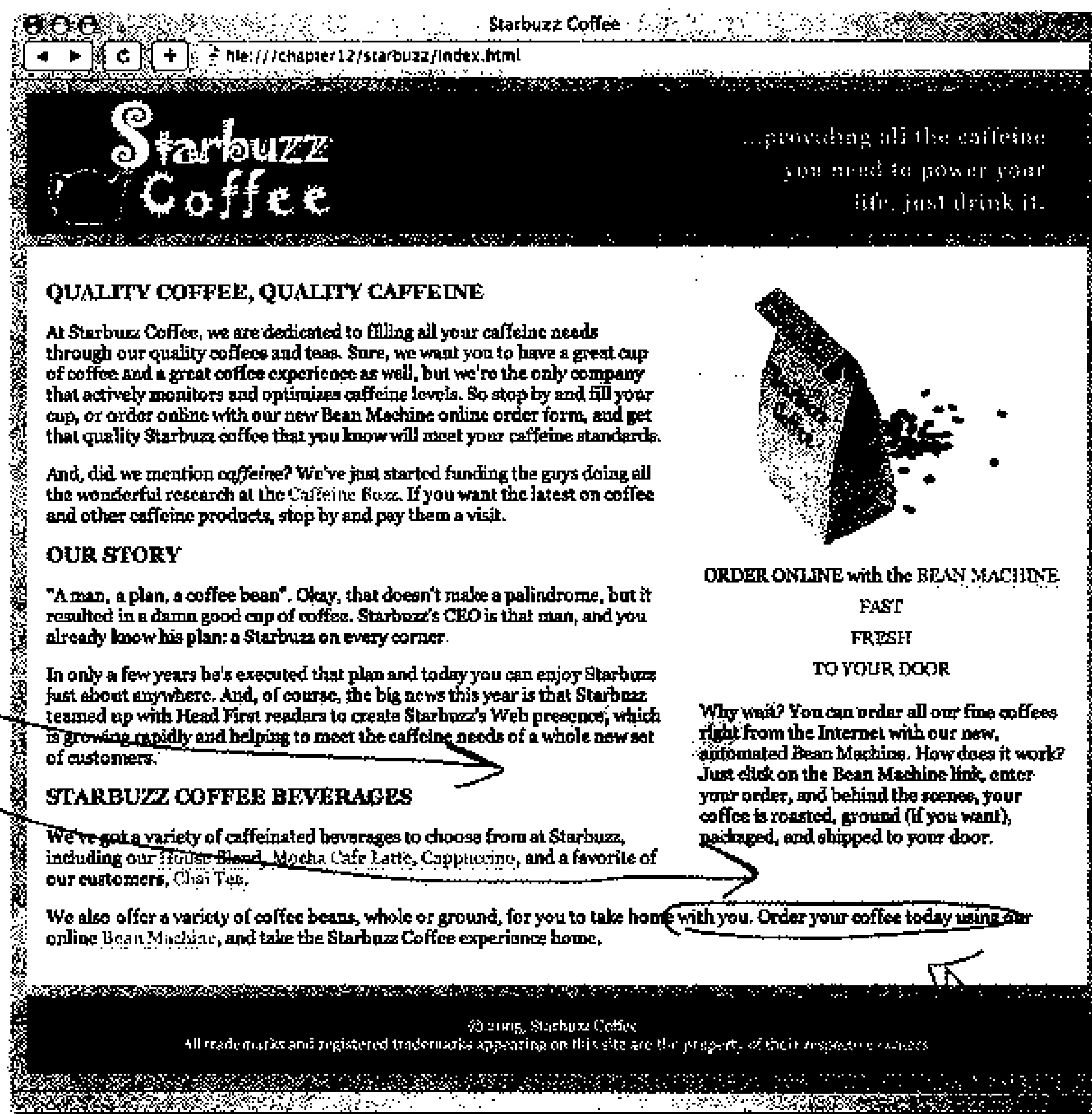
测试Starbuzz

把新sidebar属性添加到“chapter12/starbuzz”文件夹下的“starbuzz.css”文件中后，重新加载Starbuzz页面。看一下效果：

看起来非常好，但如果往前翻3页你会发现这跟我们想做的不一样。

主内容和sidebar分别在左边和右边，但实际上看起来还不像两栏。

两部分的背景图像是一样的。两栏之间没有空隙。



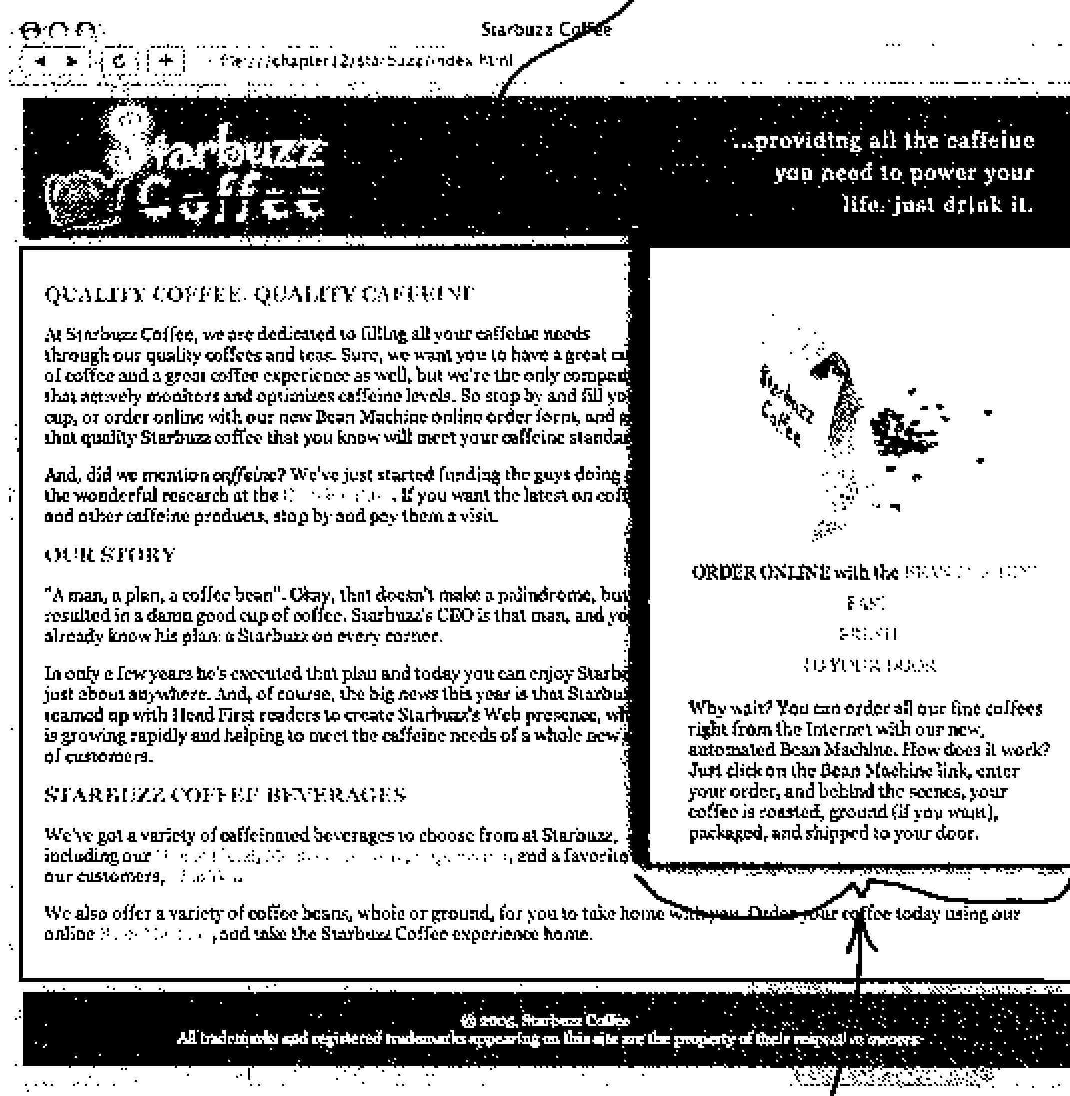
文本包围在sidebar下面，使得页面看起来一点也不像两栏。修改后的休闲室其实就是这样的——或许我们原本就希望做成这样。

搞定分栏问题

你正坐在那儿等我们骑着白马，带着能够解决一切问题的神奇属性来吗？那是不可能的。这正说明了一点，CSS中页面布局更是一门艺术——或者至少是一系列技术——而不是一系列万能的属性。所以，我们要做的是用一个普通的广泛应用的技术来解决这个问题。之后，你将看到另一些解决同样的分栏问题的方法。这儿重要的就是要理解这些技术，以及知道为什么它们有用，那么你就可以用它们解决自己的问题，甚至在需要的时候改编它们。

首先要注意的是sidebar正漂浮在页面上。主内容区一直在它下面。

如果我们给主内容区设置一个至少跟sidebar一样大的合适的边界，又会怎样呢：那么它的内容就会延伸到sidebar附近，而不是一直到它下面。



然后我们将把它们隔开，因为边界是透明的并且不显示背景图像，页面的背景颜色就会透过它显示出来。这就是我们想要的（翻到前面你会看到）。

我们把边界设置得跟sidebar一样宽。

Sharpen your pencil

我们要做的是在主内容部分设置一个与sidebar一样宽的右边界。但是sidebar有多大呢？希望你还没忘记上一章学过的内容。以下是你要计算sidebar宽度要用到的所有信息。答案在本章后。

```
#sidebar {
        background: #efe5d0 url(images/background.gif) bottom right;
        font-size:  105%;
        padding:     15px;
        margin:      0px 10px 10px 10px;
        width:       280px;
        float:       right;
}
```

你会在这条规则中找到所有计算宽度需要的信息。

设置main部分的边界

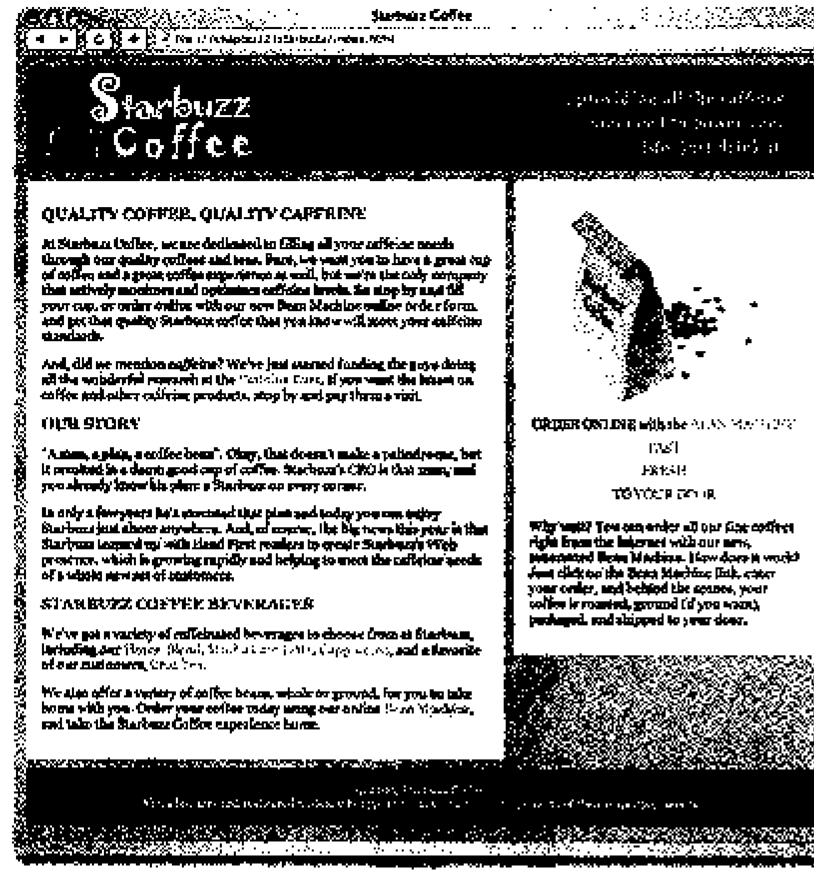
Sidebar的宽度是330像素，包括10像素的左边界，这会提供我们需要的两栏之间的空隙（出版界称之为"装订线"）。在"starbuzz.css"文件中给**#main**规则添加330像素的右边界，做法如下：

```
#main {
        background: #efe5d0 url(images/background.gif) top left;
        font-size:  105%;
        padding:     15px;
        margin:      10px;
        margin:      0px 330px 10px 10px;
}
```

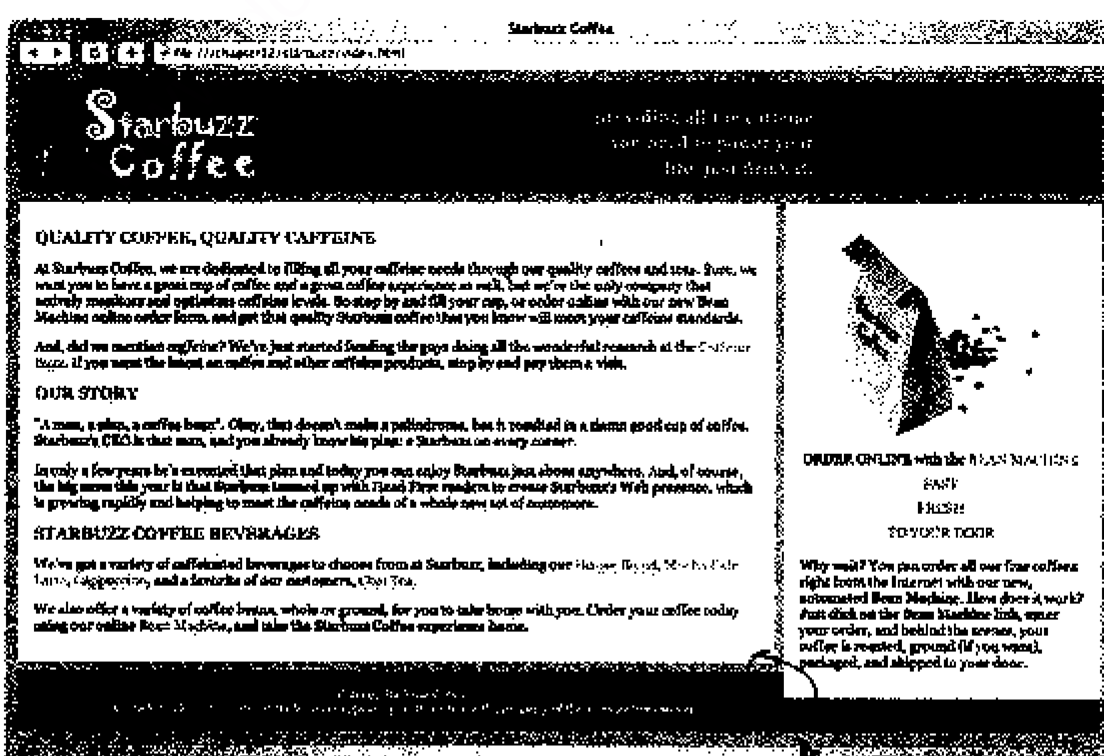
我们把右边界改为330像素，与sidebar的大小一致。

测试

跟平常一样，保存“starbuzz.css”文件并重新加载“index.html”。你现在会在两栏之间看到好看的空隙。再次思考一下这是如何实现的。Sidebar漂到右边，所以它被尽可能向右移，就把整个**<div>**从正常流中移走了，漂浮在页面的开头。现在主内容**<div>**跟浏览器一样宽（因为块元素就是这样的），但我们给它设置一个跟sidebar一样宽的边界来减小内容区的宽度。这样就形成了好看的分成两栏的外观。你知道main **<div>**的盒子仍然延伸到sidebar下面，但这只有我们知道。



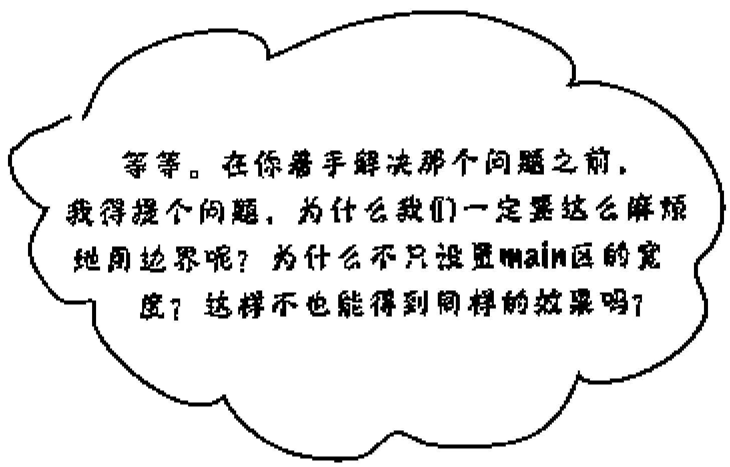
通过扩展main<div>的边界，我们造就了两栏布局的幻觉，包括之间的空隙。



这样会导致一个问题，当你把浏览器调整得更宽一些时页脚和sidebar就会重叠。

哦，还有一个问题

当你测试页面的时候你会注意到一个小问题。如果把浏览器宽度调大一些，页脚就会上升到sidebar下面。为什么会这样呢？对，记住，sidebar不在流中，所以页脚会忽略它，当内容区太短时，页脚就上升了。我们可以用同样的边界技巧设置页脚，不过那样的话页脚就只在内容区下面而不是整个页面的下面了。那该怎么办呢？

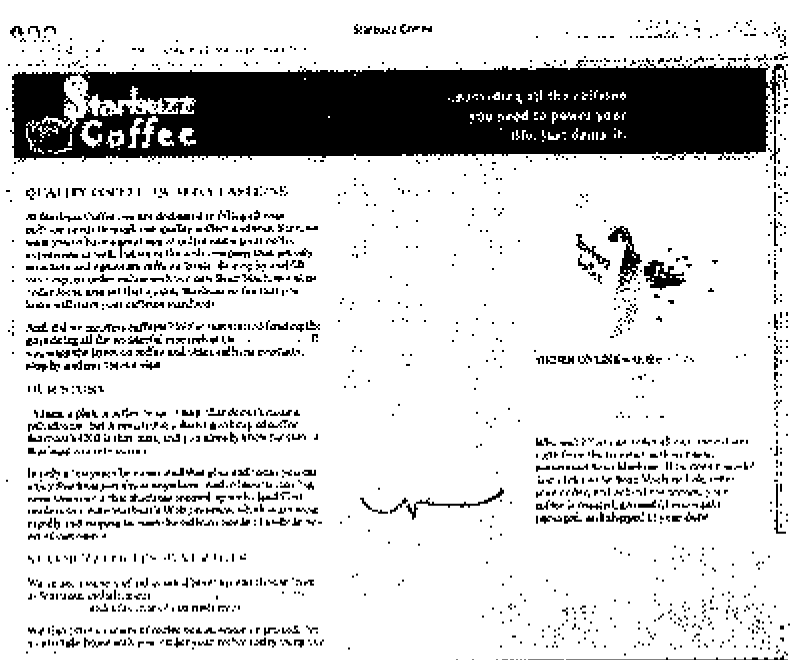




听起来不失为一个好的解决方法……先试试。

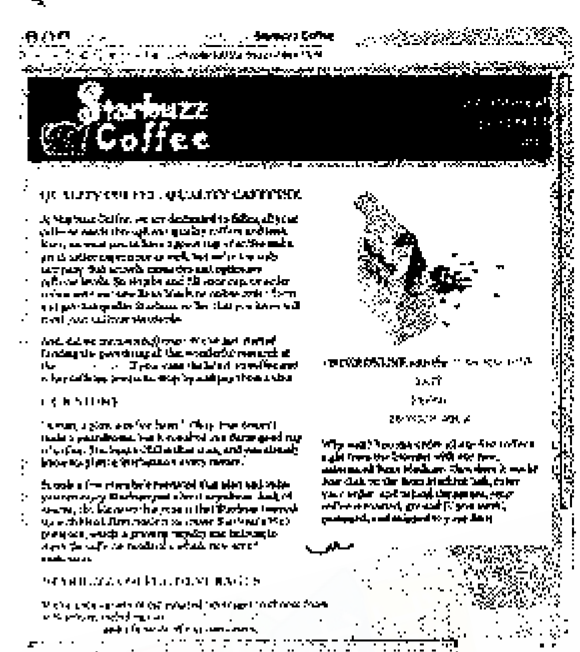
给内容区和sidebar都设置宽度就会使页面不能正确地扩展和收缩，因为它们的宽度都固定了。查看以下显示它如何工作（或者不工作）的屏幕截图。

不过这还好。你的想法不错，本章稍后在讲“流动对固定”布局的时候会重新考虑这个思路。如果我们先把其他一些内容锁定，那么将会有几种方法让你的想法实现。



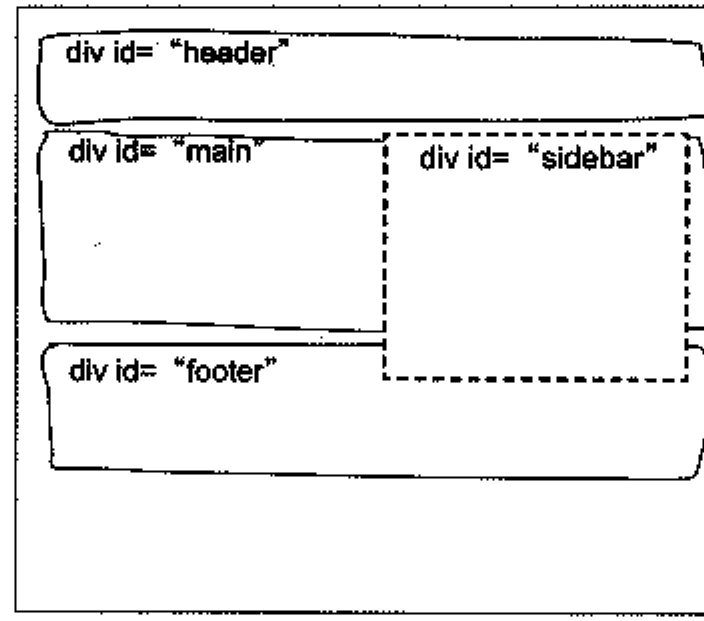
当浏览器很宽时，两部分就完全分开了。

如果把浏览器窗口变小，两部分就开始重叠。



再来处理重叠问题

猜猜会怎样，这次我们要骑着白马带着解决方案来，但最好别养成依赖的习惯。解决方案叫做**clear**属性，以下说明它是如何工作的……



现在的情况是这样的。"main" <div>缩短到footer<div>可以上升到与sidebar<div>重叠的程度。

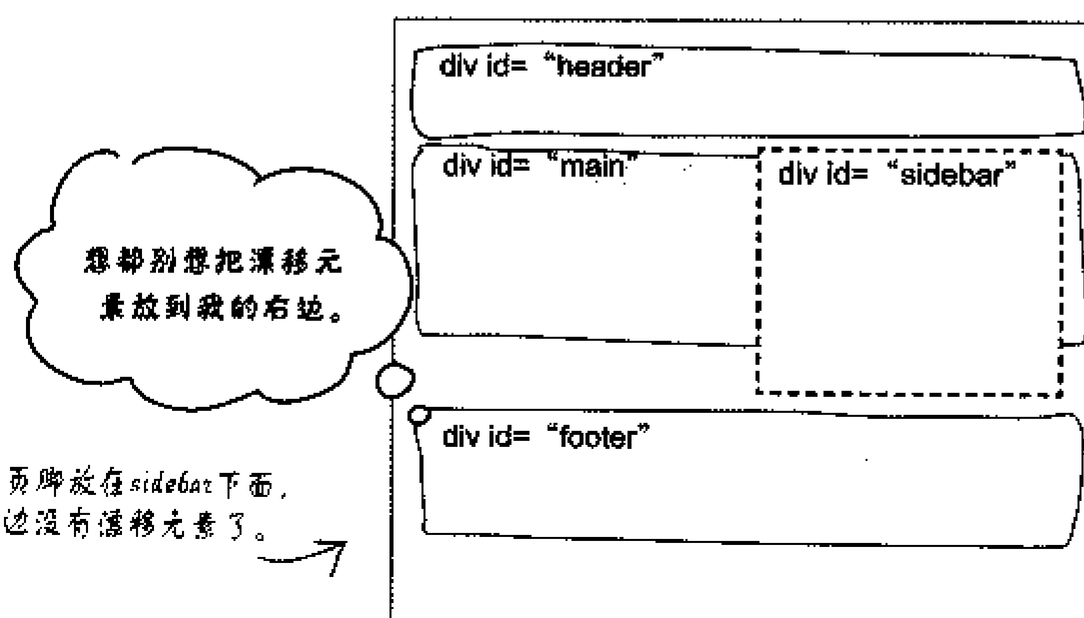
发生这种情况是因为把sidebar从流中拉出去了。所以浏览器只是把main和footer<div>正常地进行布置，忽略了sidebar（虽然浏览器流入内联元素时，它会考虑sidebar的边框并让内联元素包围在它周围）。

我们可以用CSS的**clear**属性解决这个问题。可以设置一个元素的这个属性，使元素流入页面时，不允许漂移元素出现在这个元素的左边、右边或两边。我们来试一下……

```
#footer {
        background-color:  #675c47;
        color:             #efe5d0;
        text-align:        center;
        padding:           15px;
        margin:            10px;
        font-size:         90%;
        clear:             right;
}
```

我们在这儿给footer规则添加一个属性，说明页脚的右边不能有漂移内容。

现在当浏览器在页面上放置元素时，它会注意看有没有漂移元素会到页脚的右边，如果有，它把页脚向下移动，直到它右边没有任何内容为止。现在，无论你把浏览器开得多大，页脚会一直在sidebar的下面。



现在页脚放在sidebar下面，它右边没有漂移元素了。

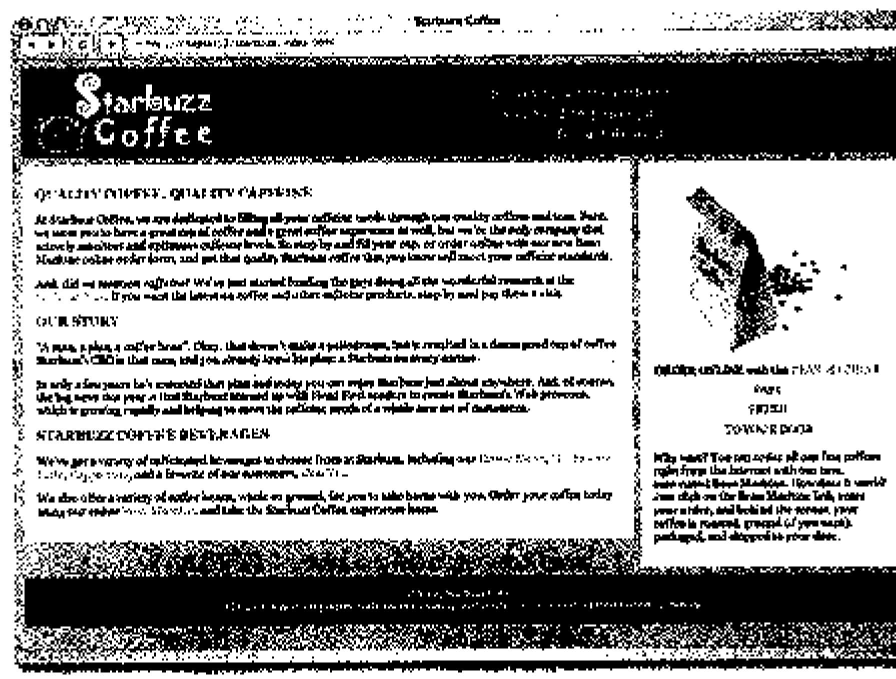
测试

试一把，把clear属性添加到"starbuzz.css"文件的footer规则中，然后重新加载"index.html"。你会发现现在即使屏幕很宽，页脚还在sidebar下面。

我们还可以考虑给这个网页做一些别的改进,比如让每栏都向下延伸到页脚。如果像现在这样，在主内容和页脚之间（如果把浏览器设置得很宽）或sidebar和页脚（如果把浏览器设置成正常宽度）之间会有空隙。不过，解决这个问题并不容易，在这章中暂不介绍。CSS中的布局是一门艺术，没有绝对完美的布局方案。如果做得好，用CSS布局会让你的网页有一个更好的外观，而且还会使页面在不支持太多CSS的浏览器中看起来也相当不错。

除了用漂移，我们还要看另外几种用CSS布置页面的方法。CSS有很多这样的方法，每种方法都有自己的缺点和优点。

现在我们的页脚问题解决了。无论浏览器是窄是宽页脚会一直在sidebar下面。



there are no Dumb Questions

问：为什么CSS中没有一个分栏属性呢？为什么让这个家伙好好工作这么难呢？

答： 哦，你赢了！你问了一个价值$64,000的问题。严格来说，似乎CSS应该有一些指定"给我两栏，该死的！"的方法，但是你必须谨记XHTML和CSS的所有意图。记住，XHTML按道理应该是一种结构和内容格式，可以用CSS样式化，即使没用CSS，也应该在任何设备中都能看得见。所以，真的不必吃惊，CSS不是文档显示的终结，我们也可能选用Microsoft Word。但是CSS的确给了你一些很棒的工具来创建有吸引力和实用的布局，如果不过于苛求的话，它还算做得不错。

问：元素能漂移到中间吗？

答： 不能。CSS只允许把元素漂移到左边或右边。但如果把元素漂移到中间，漂移元素下面的内联元素就必须流到元素的两边。如果这样做了，网页就会失去可读性和吸引力。

问： 边界会在漂移元素上相叠吗？

答： 不，它们不会，很容易看出为什么。不像流到页面上的块元素，漂移元素只是漂移。换句话说，漂移元素的边界实际上不与正常流中元素的边界相交，所以它们不会相叠。

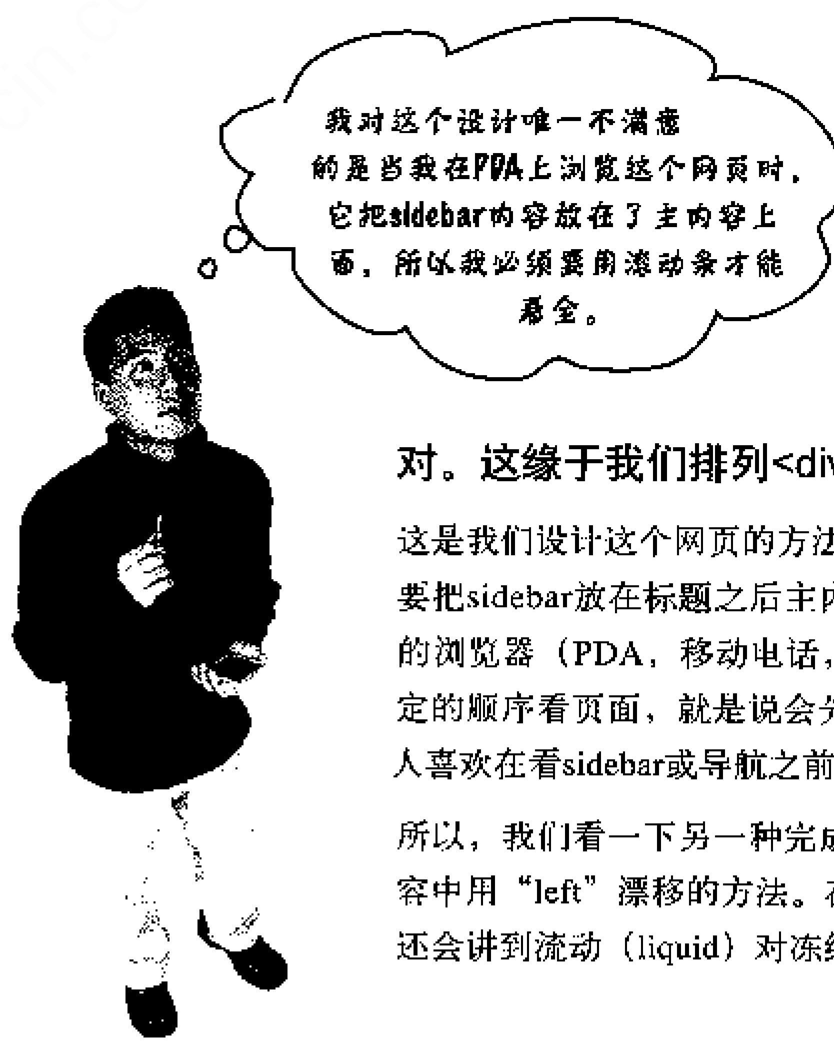
但这引出了一个重点，指出了布局中的一个普遍错误。如果有一个主内容区和一个sidebar，给每个都设置上边界是很常见的。那么，如果漂移sidebar，它还有个边界，这个边界就不会跟它前面的任何东西相叠。所以如果你不记得漂移元素和边界不重叠，可以很容易地给sidebar和主内容设置各自的边界来结束。

问： 可以漂移一个内联元素吗？

答： 当然可以。最好的例子——也是最普遍的一个——是漂移图像。试一试——在一段中把一个图像漂移到左边或右边，你会发现文本流到了它周围。别忘了添加补白来给图像一些空间，也有可能是个边框。你也可以漂移别的你喜欢的内联元素，但这并不常见。

问： 块元素忽略漂移元素，而内联元素知道它们在那儿，对吗？

答： 是的，这是个理解它的好方法。嵌套在块元素中的内联元素通常流到漂移元素周围，留意漂移元素的边界，而块元素跟平常一样流向页面。例外的是当你在一个块元素中设置了clear属性，它会把一个块元素向下移直到它的右边、左边或两边（取决于clear的值）都没有漂移元素。



对。这缘于我们排列<div>的顺序。

这是我们设计这个网页的方法的缺点之一——因为我们需要把sidebar放在标题之后主内容之前，任何使用功能有限的浏览器（PDA，移动电话，屏幕读取器等）的人会以固定的顺序看页面，就是说会先看到sidebar。然而，大多数人喜欢在看sidebar或导航之前看到主内容。

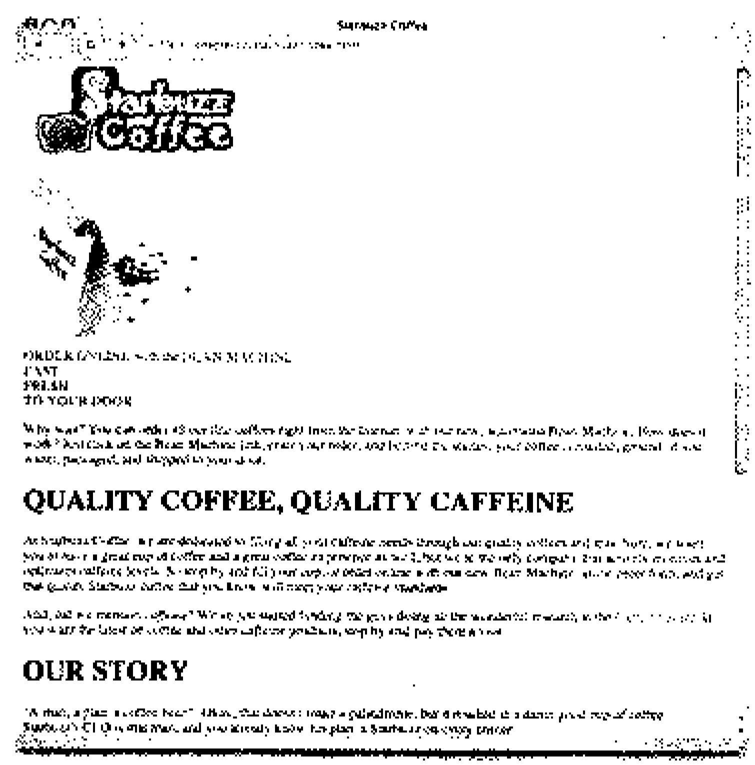
所以，我们看一下另一种完成任务的方法，也就是在主内容中用“left”漂移的方法。在我们解释这个的时候，我们还会讲到流动（liquid）对冻结（frozen）设计。

练习

看，没有CSS！

想知道如果你的用户处在条件很差的情况下（比如在一个不支持CSS的浏览器上）你的网页会有什么样的外观吗？那么打开“index.html”文件并从<head>中移走<link>，保存，并在浏览器中重新加载页面。现在你可以看到内容显示的真正顺序了（或者从屏幕阅读器那儿听说）。动手试试吧，完成后一定要把它还原（毕竟，这一章是关于CSS的）。

这是没有CSS的Starbuzz页面，大部分处于良好状态。还是很具可读性的，尽管Bean Machine在主内容之前出现了（这可能不是我们想要的）。



右紧左松

我们把Starbuzz页面转变一下以使主内容漂移到左边。我们会仔细观察它如何运作，然后继续使它真正工作。你会发现记忆的右紧左松在CSS世界中也适用……所以无论如何，对于我们的sidebar是适用的。以下是我们转换网页的方法——只用简单的几步。

第一步：从sidebar开始

我们基本上只是转换一下sidebar和主内容区的角色。主内容区将有一个固定的宽度和漂移，而sidebar将分布在内容周围。我们也要用同样的边界技术让两者保持视觉上的分离。不过在我们开始改变CSS之前，找到“index.html”文件并把“sidebar” **<div>**移到“main” **<div>**下面。完成之后，在sidebar 的CSS规则中做以下改动：

```
#sidebar {
        background:  #efe5d0 url(images/background.gif) bottom right;
        font-size:   105%;
        padding:     15px;
        margin:      0px 10px 10px 470px;
        width:       280px;
        float:       right;
}
```

（width: 280px; 与 float: right; 两行已被划掉。）

因为现在sidebar要流到主内容下面，我们需要把大边界传给sidebar。主内容区的总宽度是470像素。（空闲的时候用和sidebar一样的方法自己算一下宽度。我们要把主内容区的宽度设置为420像素。）

我们把主内容<div>设置成了固定宽度，所以把sidebar 的宽度属性和漂移值删除。

第二步：关注主内容

现在我们需要漂移main**<div>**，以下是做的方法：

```
#main {
        background: #efe5d0 url(images/background.gif) top left;
        font-size:    105%;
        padding:      15px;
        margin:       0px 10px 10px 10px;
        width:        420px;
        float:        left;
}
```

我们把右边界从330像素改回10像素。

我们需要设置一个确定的宽度，因为我们要漂移这个元素。我们就用420像素。

我们将把main<div>漂移到左边。

第三步：小心页脚

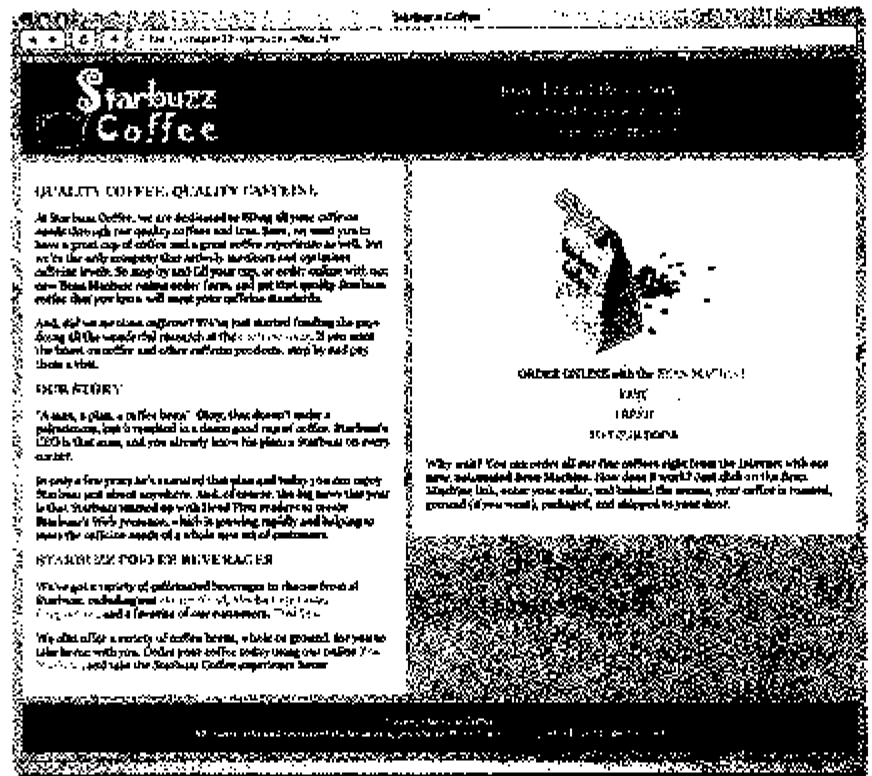
现在，我们只需要footer来把所有东西清理到左边，而不是右边。

```
#footer {
        background-color:  #675c47;
        color:             #efe5d0;
        text-align:        center;
        padding:           15px;
        margin:            10px;
        font-size:         90%;
        clear:             left;
}
```

把clear属性值改成left，而不是right，那样footer照样会清理主内容区。

快速测试

我们已经说过这种把内容漂移到左边的方法会有一些问题。在继续进行之前做一个快速测试来看一下这些都如何工作。试一下，在“starbuzz.css”文件中做改动，然后把“index.html”重新加载到浏览器中。熟悉一下当把浏览器调整到窄、正常和宽时，页面如何显示。



实际上，这看起来非常好，那些<div>现在的顺序很正确。Sidebar通常被用于导航，扩展的话看起来不是很好。固定下来会比较好。

当我们把sidebar<div>漂移到右边，设计就显得漂亮又紧凑，内容也可以扩展，但如果我们把主内容漂移到左边，并允许sidebar扩展的话，设计就显得太松散了。

BRAIN POWER

从设计的角度考虑，第一种方法较好，而从信息的角度考虑，第二种方法较好（因为<div>的位置）。有没有一种方法可以把两种的优点结合起来：让sidebar的宽度固定下来，而main<div>还在XHTML开头？我们可以做什么折衷设计来达到这个目的呢？

流动和冻结设计



迄今为止，我们做的设计都叫做流动布局，因为无论我们把浏览器调整到多大，它们都会扩展到填满浏览器为止。这些布局很有用，因为通过扩展，它们把现有的空间填满了，并且允许用户充分利用屏幕空间。不过，有时把布局锁定很重要，以便当用户调整屏幕时，你的设计看起来还跟原来的一样。有几个布局是这样工作的，我们先从冻结布局开始。冻结布局把元素锁住，冻结到页面上，所以它们根本不能移动，因此我们避免了许多窗口扩展带来的问题。我们来试一下冻结布局。

从你现在的页面到冻结布局只需要给XHTML增加一样东西，再给CSS增加一条新规则。

改变XHTML

在你的XHTML中你要添加一个新的id为“allcontent”的**<div>**元素。如它的名字所暗示的，这个**<div>**要包围页面中的所有内容。所以把开始**<div>**标签放到标题**<div>**之前，结束标签放在footer **<div>**之后。

```
<body>
   <div id="allcontent">
      <div id="header">
        ... rest of the XHTML goes here ...
      </div>
   </div>
</body>
```

添加一个新的id为“allcontent”的<div>元素包围<body>中的所有其他的元素。

这个<div>结束footer<div>。

改变CSS

现在你要用这个**<div>**来把所有元素的大小和“allcontent”**<div>**中的内容限制为800像素的固定宽度。以下的CSS规则可以做到这些：

我们要把“allcontent”的宽度设置为800像素。这样的效果是把所有它包含的内容限制在800像素以内。

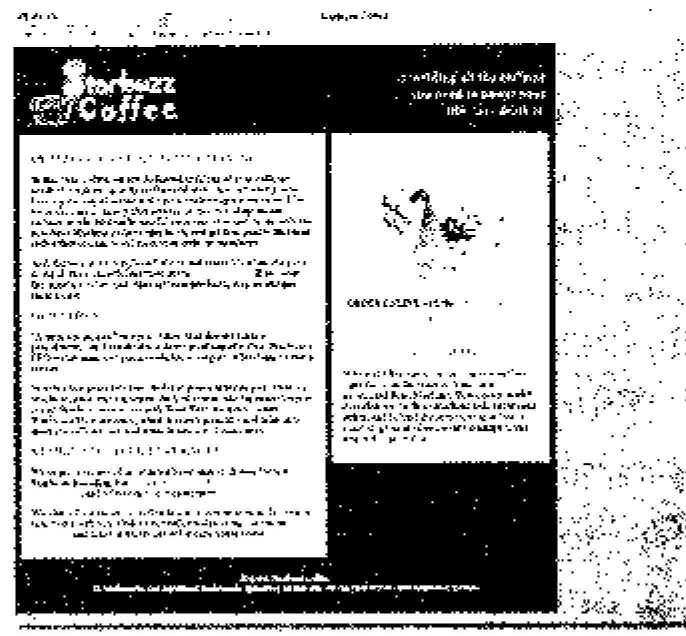
```
#allcontent {
        width:             800px;
        padding-top:       5px;
        padding-bottom:    5px;
        background-color:  #675c47;
}
```

我们做这些的时候，因为是第一次样式化这个<div>，可以添加一些补白并且给它设计自己的背景颜色。你会发现这有助于把整个页面约束在一起。

外部的“allcontent”<div>通常是800像素，甚至当浏览器被重新调整的时候也是这样，所以我们有效地把<div>以及它包含的所有内容都冻结到页面上了。

冻结布局测试

继续把这条规则添加到“starbuzz.css”末尾，然后重新加载“index.html”，现在你可以看到为什么我们叫它冻结布局了。当调整浏览器时它是固定不动的。



现在无论你如何调整浏览器，“allcontent” <div>的宽度都是800像素，因为其他的<div>都在“allcontent”里，它们都会调整到800像素的空间，所以页面基本上冻结为800像素。

这当然解决了sidebar扩展的问题，看起来很不错。浏览器很宽的时候稍微有些奇怪，因为所有空的空间都在右面。

但是，我们还没完成任务，还有一个地方要改。



流动的和冻结的之间的状态是什么？当然是凝胶物！

冻结布局有几个好处，但当你把浏览器调宽时外观明显变差。不过我们有一个解决方法，它是一个你能在网站上看到的很普通的设计。这种设计在冻结的和流动的之间，它有一个相应的名字叫凝胶物。凝胶物布局锁定了页面中内容区的宽度，但把它放到浏览器中央。其实让你实际操作把布局改为凝胶物布局比我解释它如何显示页面要简单，所以我们来做吧：

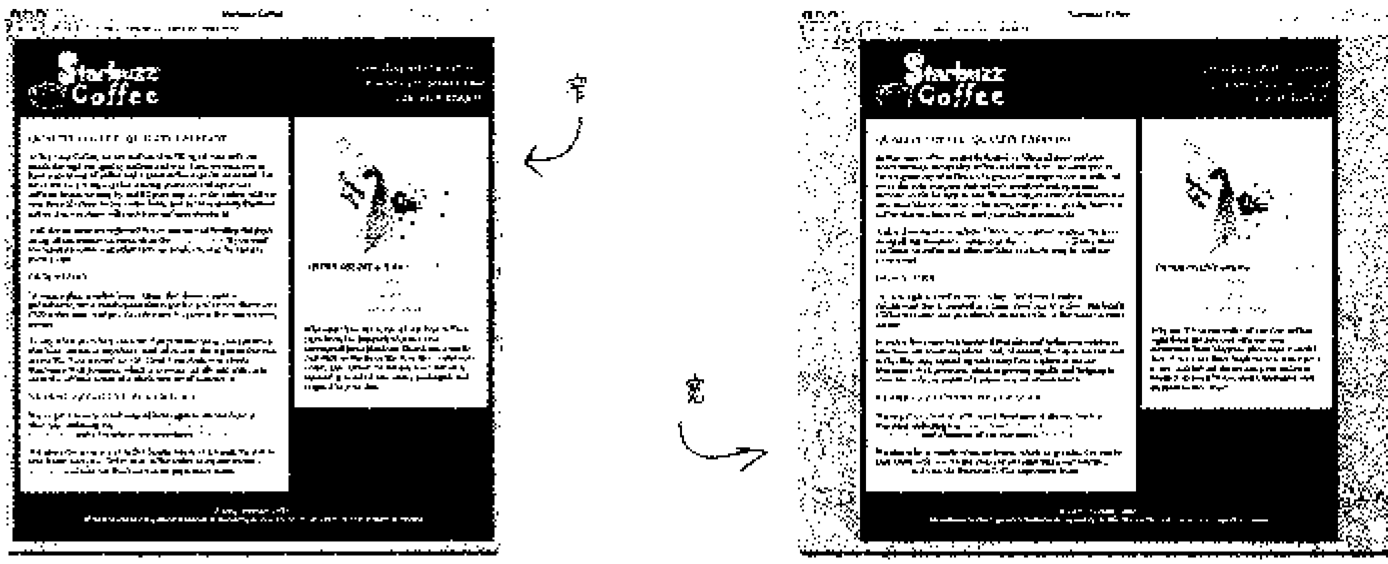
```
#allcontent {
        width: 800px;
        padding-top:     5px;
        padding-bottom:  5px;
        background-color: #675c47;
        margin-left:  auto;
        margin-right: auto;
}
```

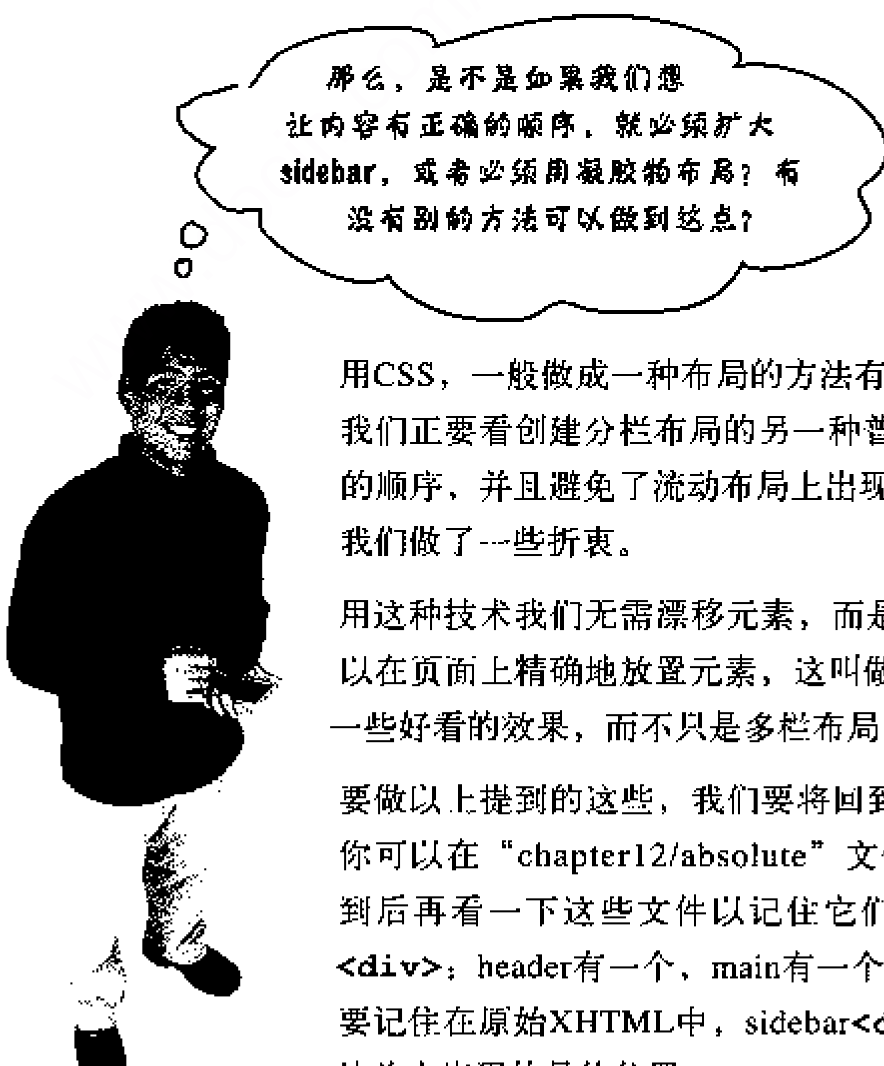
不是把“allcontent” <div>的左右边界固定，而是把边界设置成“auto”。

你是否还记得我们讲过，当把一个内容区的宽度设置为“auto”时，浏览器根据内容区的需要调整内容区的宽度。把边界设置成“auto”，浏览器会计算出合适的边界是多少，并且保证左右边界相等，以使内容居中。

测试凝胶物布局

把两个边界属性添加到你的“starbuzz.css”文件，然后重新加载页面。
现在调整浏览器窗口的大小。非常不错，对吗？





用CSS，一般做成一种布局的方法有好几个，都有各自的优缺点。实际上，我们正要看创建分栏布局的另一种普通的技术，它也可以让内容保持正确的顺序，并且避免了流动布局上出现的一些问题。不过，如你将要看到的，我们做了一些折衷。

用这种技术我们无需漂移元素，而是要用到CSS的一个特点，就是用它可以在页面上精确地放置元素，这叫做绝对布置。你也可以用绝对布置制造一些好看的效果，而不只是多栏布局。我们也会看一个这样的例子。

要做以上提到的这些，我们要将回到这章开头出现的原始XHTML和CSS。你可以在“chapter12/absolute”文件夹下找到这些文件最初的复件。找到后再看一下这些文件以记住它们原来的样子。回想一下我们有一堆`<div>`：header有一个，main有一个，footer有一个，sidebar也有一个。还要记住在原始XHTML中，sidebar`<div>`紧跟在主内容区后面，这是我们认为它出现的最佳位置。

绝对布置如何工作

我们从了解什么是绝对布置，以及它如何工作开始。这儿有一小段用绝对布置放置sidebar**<div>**的CSS。暂时不要把这段打进去，现在我们只是想让你熟悉一下它是如何工作的：

我们做的第一件事是用position属性指定元素应该被绝对放置。

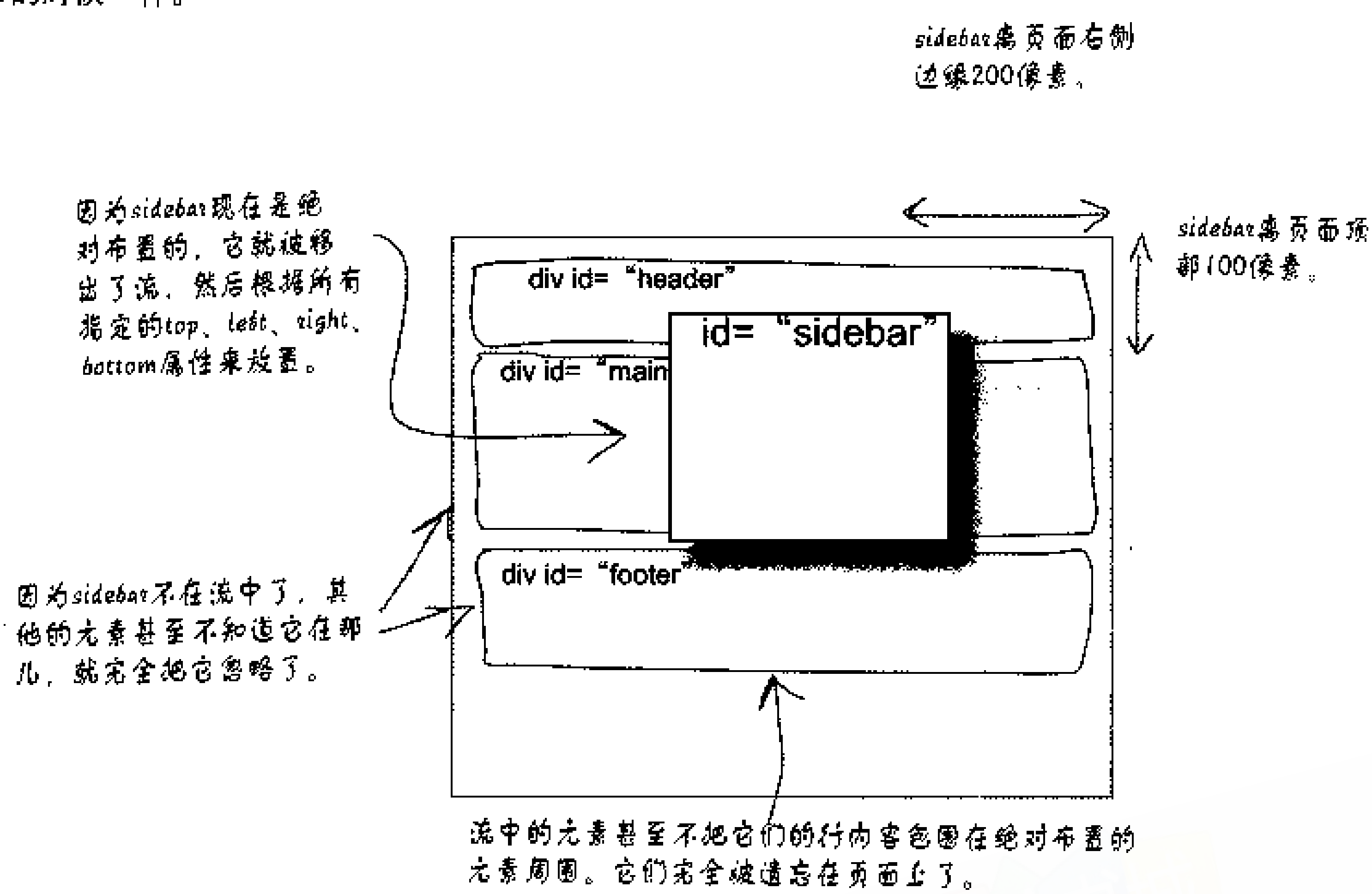
```
#sidebar {
    position: absolute;
    top:    100px;
    right:  200px;
    width:  280px;
}
```

接着我们设置top和right属性。

同时还设置<div>的width。

CSS做什么

现在我们来看看这段CSS做什么。当一个元素被绝对放置了，浏览器做的第一件事是把它从流中完全移走。接着浏览器把元素放置在**top**和**right**属性指明的位置（你也可以用**bottom**和**left**）。在这个例子中，sidebar离页面顶部100像素，离页面右侧边缘200像素。我们还设置了这个**<div>**的width，与它被漂移的时候一样。



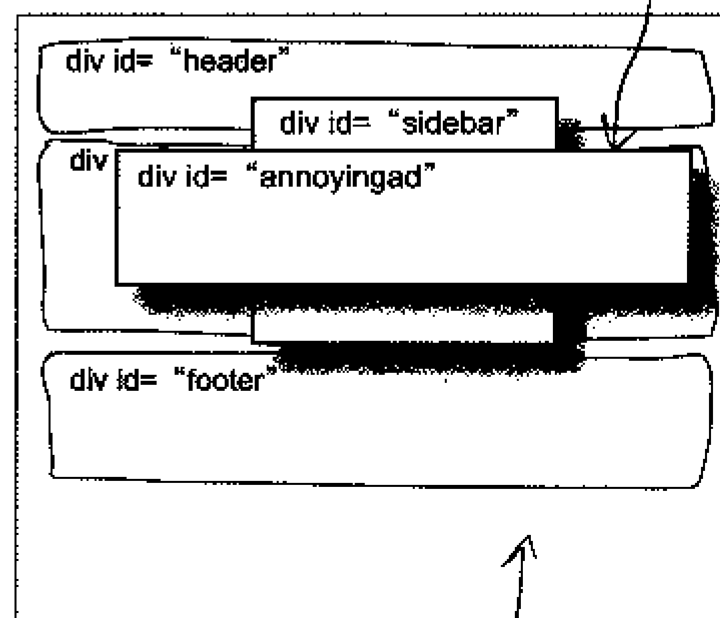
绝对布置的另一个例子

我们来看另一个例子。假设另外有一个**id**为“annoyingad”的**<div>**。我们可以这么放置它：

```
#annoyingad {
    position: absolute;
    top:    150px;
    left:   100px;
    width:  400px;
}
```

annoyingad被放置在离左侧100像素，离顶部150像素的位置。它比sidebar稍微宽点，是400像素。

现在我们有了第二个绝对布置的<div>，大约<div>离左侧100像素，离顶部150像素。



注意，annoyingad <div>在sidebar<div>上方。

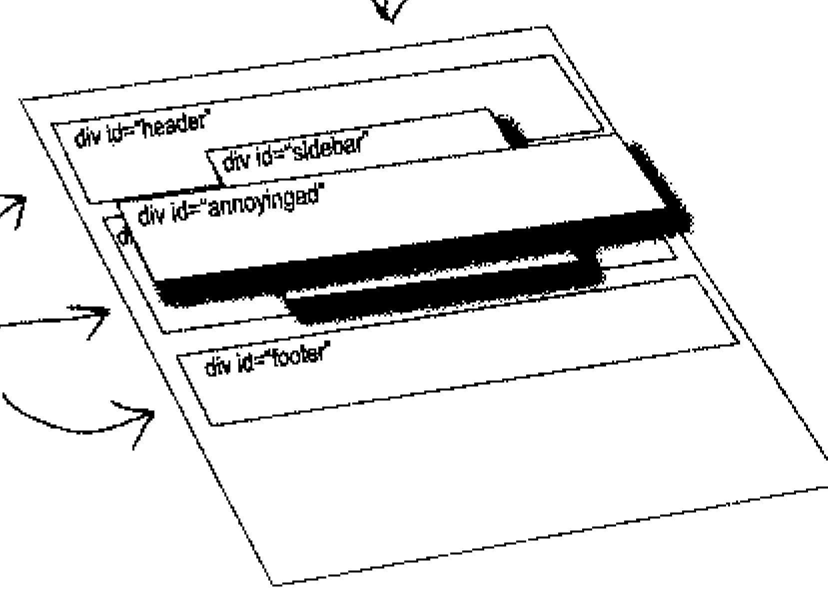
与sidebar一样，我们把annoyingad **<div>**放在页面上的一个精确位置。它后面的任何元素都在页面的正常流中，与上层绝对布置的元素没有任何联系。这跟漂移一个元素稍微有些不同，因为流中的元素要调整它们的行内容还要考虑到漂移元素的边界。但绝对布置的元素对其他元素没有任何影响。

谁在上面？

关于绝对布置的元素的另一点有趣的地方是，你可以把它们互相层叠放置。但如果你把几个绝对布置的元素放在一个页面的同一位置，你怎么知道层次呢？换句话说，谁在谁上面？

每个布置的元素有一个叫做**z-index**的属性，用来指定它的层叠位置。在后面几页中你就会看到如何定义一个z-index。

annoyingad <div>和sidebar <div>都在页面上，annoyingad 有一个较大的z-index值，因此它在上面。



header、main、footer <div>都在流中，并且在页面的同一个平面上。

there are no Dumb Questions

问：默认的positon属性值是什么？

答： 布局的默认值是“static”，static布局把元素放置在正常的文档流中，并且不用你布置——浏览器决定哪个元素放哪儿。你可以用float属性把一个元素从流中拿出来，可以决定它应该漂向左边还是右边，不过最终还是浏览器决定它放置的位置。与之相对的是positon的属性值“absolute”，用绝对布置，你告诉浏览器放置元素的精确位置。

问：我可以只布置<div>吗？

答： 你可以布置任何元素，无论是块元素还是内联元素。只是记住当绝对放置一个元素时，就把它从正常的页面流中移走了。

问：所以，我可以布置内联元素？

答： 是的，你当然可以。例如，布置<img>元素是很常见的。也可以布置<em>，<span>及其他，但一般不这么做。

问：positon属性值除了static和absolute还有其他的吗？

答： 实际有四个：static，absolute，fixed和relative。你已经听说过static和absolute。fixed把一个元素放置在跟浏览器窗口（而不是页面）有关的位置，所以fixed元素不移动。在后面几页中你就会看到一个Fixed的例子。relative取出一个元素并让它正常地流到页面上，然后在页面显示之前让它偏移。relative多用于比较高级的布局和特殊的效果。你将会看到一个这样的例子。

问：我必须要与漂移元素一样指定绝对布置元素的宽度吗？

答： 不，你不一定非要指定绝对布置元素的宽度。如果你没指定，块元素默认状态下会占据整个浏览器的宽度，减去你指定的左边或右边的偏移。这可能正是你想要的，或者不是。所以如果你想改变这种默认情形的话就设置宽度属性值。

问：我是不是必须要用像素指定位置？

答： 不，还有另外一种常用的布置元素的方法，即百分数。如果你用百分数，元素的位置会随着浏览器宽度的改变而改变。例如，如果浏览器窗口宽度是800像素，元素的左位置设置成10%，那么就离浏览器窗口的左边80像素。但如果把浏览器窗口调整到400像素宽，宽度就会减小到400像素的10%，既离浏览器窗口的左边40像素。

百分数的另一个常用的用途是指定宽度。如果不需要元素或边界具体的宽度，那么可以用百分数把主内容区和sidebar的大小设置得灵活一些。你会发现在两栏和三栏布局中这种用法贡献很大。

问：要用绝对布置，我是不是必须知道如何使用z-index？

答： 不，z-index似乎大多被用于各种高级的CSS使用中，特别是在包括网页脚本的时候，所以它们稍微有些超出这本书的范围。但是它们是绝对布置工作方法的一部分，知道z-index是很有用的（事实上，你将会意识到它的重要性）。

使用绝对布置

我们现在要用与漂移版本一样的技术来创建两栏的Starbuzz页面，不过，这次要用绝对布置。以下是你要做的：

1. 首先，我们要将sidebar<div>绝对布置。事实上，我们要把它准确地放在先前我们漂移的相同位置。

2. 接着，我们要给主内容设置另一个大边界，以便sidebar可以布置在边界空间上。

3. 最后，我们要仔细测试一下这个新页面，看它与漂移版本相比有什么不同。

改变Starbuzz CSS

记住，我们要回到文件的最初版本，你可以在“chapter12/absolute”文件夹下找到。

XHTML已经准备好要工作了，sidebar<div>也刚好在我们希望的地方（重要的主内容之后）。我们需要做的只是在CSS中做一些改动，把sidebar变成绝对布置的。打开“starbuzz.css”文件，给sidebar做一些改动：

可以在“absolute”文件夹下解决，或者像我们一样把文件“index.html”和“starbuzz.css”复制到“starbuzz”文件夹中。

```
#sidebar {
    background: #efe5d0 url(images/background.gif) bottom right;
    font-size:  105%;
    padding:    15px;
    margin:     0px 10px 10px 10px;
    position:   absolute;
    top:        128px;
    right:      0px;
    width:      280px;
}
```

好了，现在我们要指定sidebar绝对布置在离顶部128像素，离页面右边0像素。我们还想让sidebar有个宽度，所以把它设置成与漂移版本一样：280像素。

马上你就会看到“128”是从哪儿来的……

离右边0像素确保sidebar紧挨浏览器的右面。

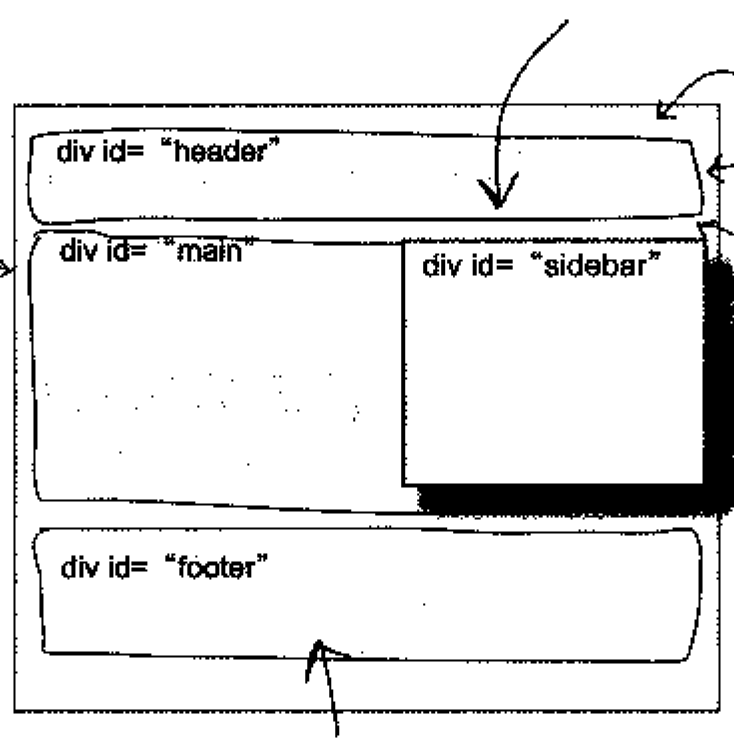
我们现在只需改动main<div>

其实没什么大改动可做，只要像在漂移版本中一样添加一个边界就可以了。那么，把main**<div>**的右边界改为330像素，与上次做的一样。

```
#main {
    background: #efe5d0 url(images/background.gif) top left;
    font-size:  105%;
    padding:    15px;
    margin:     0px 330px 10px 10px;
}
```

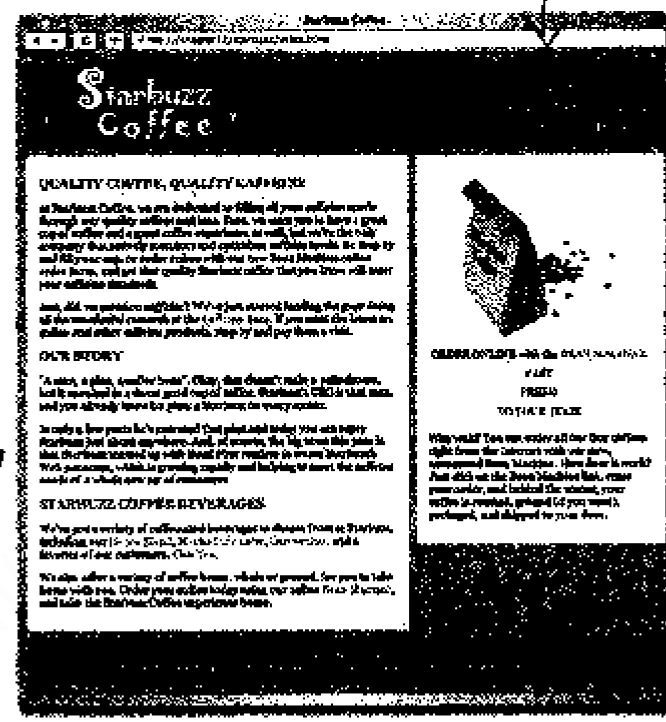
我们通过给main<div>设置一个大边界，给了一些放置sidebar的空间。其实与在漂移中用的技术一样，唯一的不同是sidebar<div>浮在边界上。

好了，你需要做的只是改变边界并保存。不过在测试之前，先考虑一下绝对布置的sidebar的布局是如何工作的。

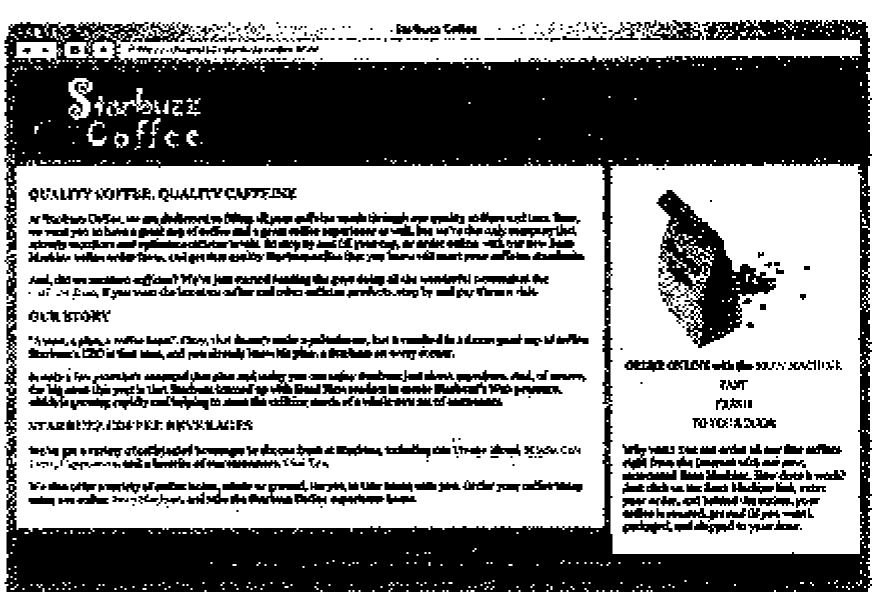


测试绝对布置

务必保存新的CSS，然后重新加载“index.html”到浏览器，看一下结果：



但是页脚还是有问题。当浏览器变得很宽时，绝对布置的sidebar会向下盖到页脚上面。不幸的是，这次我们不能求助于**clear**属性，因为流元素忽视了绝对布置元素的存在。



怎么办？有没有一种创建不易被破坏的两栏布局的方法？

好了，你知道CSS背后的动机之一是把结构和样式分离开，对吗？CSS做了一项伟大的工作，用它可以创建能在许多不同的浏览器（甚至是屏幕读取器或文本浏览器）中使用的XHTML，而且不必在这些XHTML中嵌入不必要的样式。不过这也意味着CSS不会是一种完美的页面布局语言。然而，它提供了一些有趣的工具，你可以用它们在XHTML文档中安排和布置元素。你的布局可能会出现不同的结果，这取决于页面被浏览的环境。如果把浏览器调整到极宽，那么，布局就会被破坏。

现在，你掌握了多少？在这章中我们看了几种创建两栏布局的技术。其中没有一种是完美的，事实上，它们都有许多折衷。我们快速浏览一下这几个例子。

漂移布局

哦，多可爱啊，还记得你的第一个两栏Starbuzz页面吧？你用了一个**float**属性，页脚中用了一个**clear**，一切都很好。唯一的问题是这种解决方法会导致内容放置顺序的改变，如果你的用户使用另一种浏览器，比如大声给用户读内容的屏幕读取器，它们也许不会喜欢这种顺序。

凝胶物布局

首先我们通过把页面中所有的内容用固定大小的**<div>**包围来创建了一个冻结布局，然后通过用“auto”属性值让边界扩展来做成凝胶物布局。这种方法创造出了外观美妙的布局，网上许多页面都用这种设计；这也解决了内容顺序的问题。这种方法的缺点是内容不会随着浏览器窗口的变化而扩展（不过许多人根本不没有发觉这个缺点）。

绝对布局

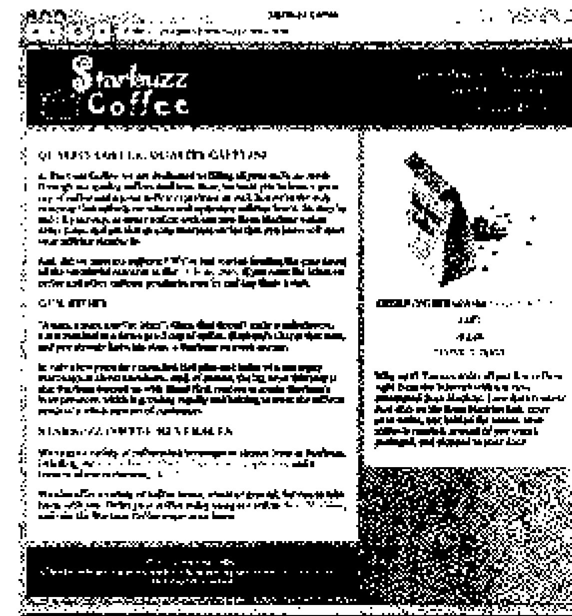
最后，我们有一个任务是创建一个流动布局，同时内容顺序是我们想要的。所以，我们用了绝对布置并且实际上达到了我们的目标。不过，还是有个缺点：因为绝对元素不能用**clear**属性，浏览器变宽时页脚蔓延到了sidebar下面。

那么我们完成了吗？也许是吧。如果这些设计中有一个能满足你的需要，很好，就用它吧。例如，许多人很高兴用凝胶物布局。不过你总是还能做一些别的调整来使你独特的布局更完美。例如，用绝对设计。我们能把页脚固定吗？倒不是真的能，不过我们可以做一个折衷。如果页脚只在主内容区之下显示的话你的设计也许会更好一些。这种情况的话，我们可以搞定页脚问题。我们来快速试一下。

固定页脚的折衷办法

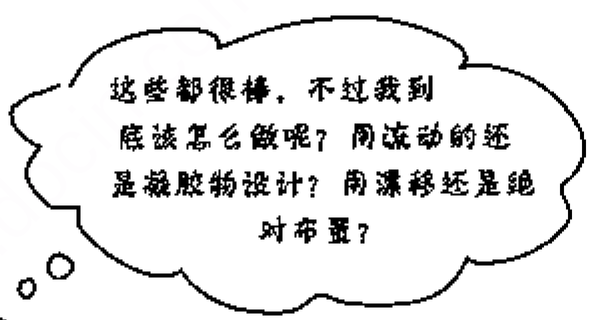
要试试这种解决方法，只需把页脚的右边界大小设置成与主内容区的右边界大小一样，像这样：

```
#footer {
        background-color:  #675c47;
        color:             #efe5d0;
        text-align:        center;
        padding:           15px;
        margin:            10px 330px 10px 10px;
        font-size:         90%;
}
```



这个布局中页脚在整个页面中处于主内容区的下面，避免延伸到sidebar的下面。

如果你保存并重新加载页面，你会看到页脚只在主内容区之下，再也不会蔓延到sidebar之下。这是最理想的吗？不，不过也不差。正如我们所说过的，CSS布局是一门艺术。要做一个布局，你需要实验，探索，并密切关注别人用CSS创建的布局（在本书结尾你会发现一些很好的CSS用法参考）。

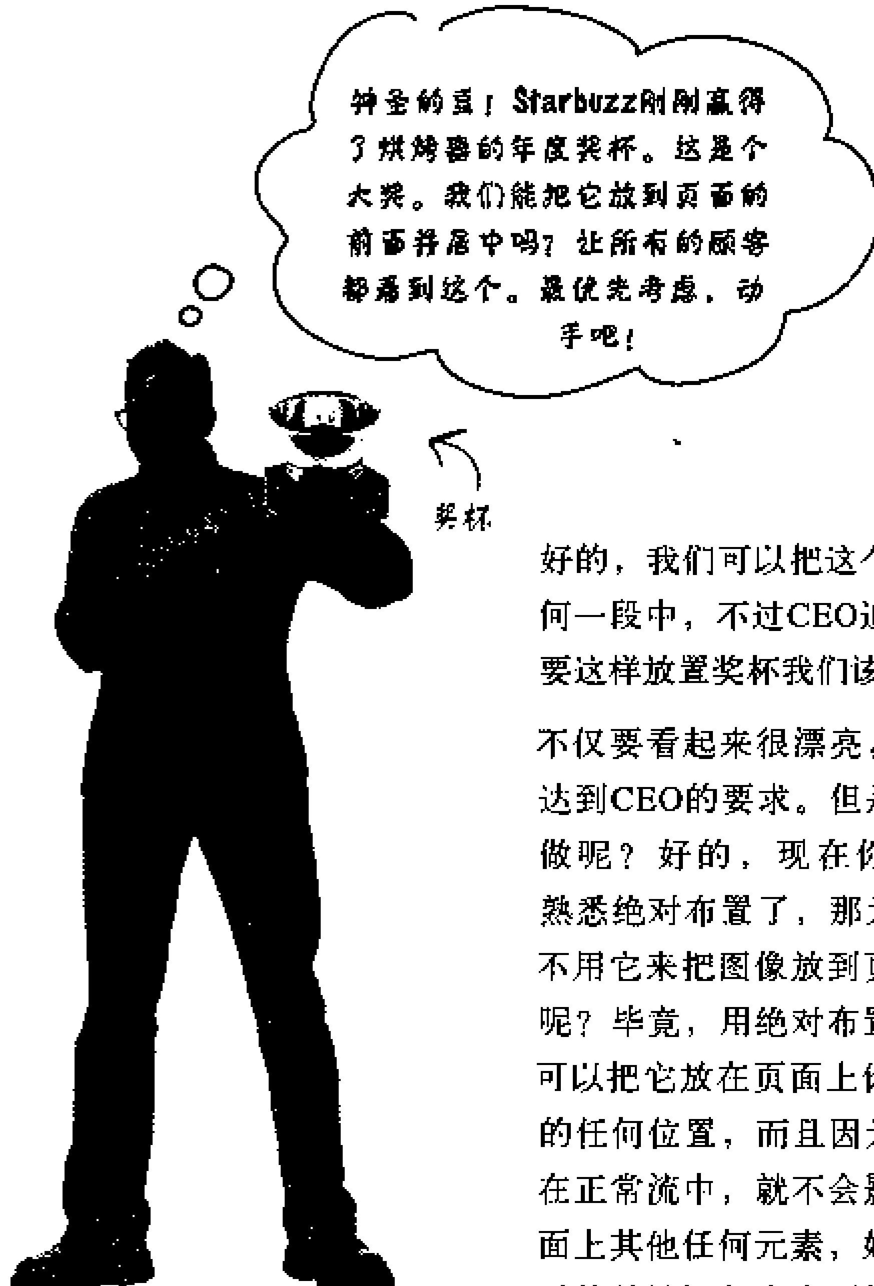




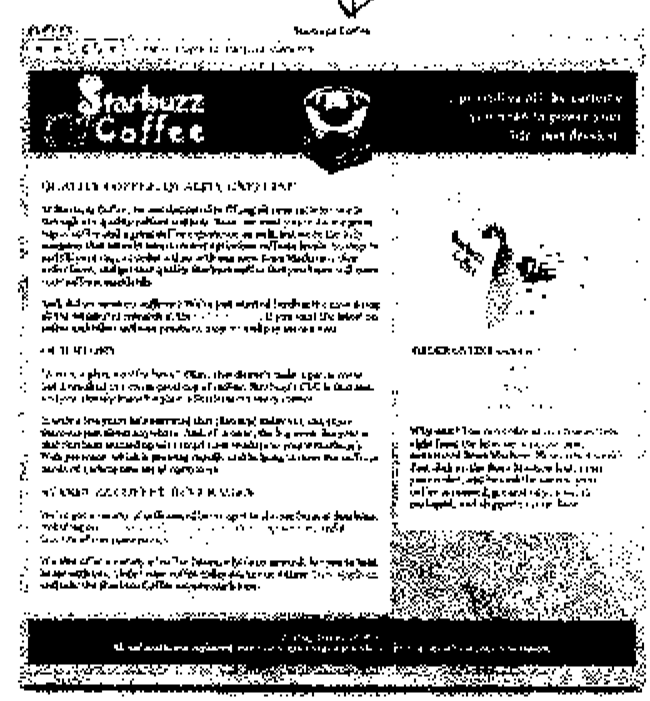
对于你想让你的布局成为流动的还是冻结的还是凝胶物的，这其实是一个决定哪种最有利于你的页面工作的问题。对某些页面来说，固定的内容区加上可扩展的边界工作得不错，并且在很宽的浏览器中也更具可读性。其他用途方面，你也许想应用于尽可能多的浏览器。所以，仔细考虑哪种布局最适合你。

一旦你决定了，你还要指出要用哪种方法创建你的页面（漂移？绝对？结合几种？）。你已经学习了基础，所以现在是开始探索的时候了，因为还有许多其他方法，而且每天都有新的被创造出来。你从这章中学到的技术通常作为比较复杂的设计的基础。

你应该知道在一般情况下，多栏布局中用漂移被认为是最灵活的解决方法。谨记，你必须小心内容的顺序，它取决于设计。



好的，我们可以把这个当作图像放到页面上已经有的任何一段中，不过CEO迫切想让它成为页面的焦点。如果要这样放置奖杯我们该怎么做呢？



不仅要看起来很漂亮，还得达到CEO的要求。但是怎么做呢？好的，现在你已经熟悉绝对布置了，那为什么不用它来把图像放到页面上呢？毕竟，用绝对布置，你可以把它放在页面上你想放的任何位置，而且因为它不在正常流中，就不会影响页面上其他任何元素，好像只需简单地添加点东西就可以了，而不用破坏已有的东西。

试试吧！从添加一个新的**<div>**开始，正好在标题下（CEO觉得这非常重要，所以它应该在主内容之前）。以下是**<div>**

```
<div id="award">
    <img src="images/award.gif"
         alt="Roaster of the Year award" />
</div>
```

<div>包含了奖杯图像。

放置奖杯

当浏览器展开为800像素（标题中图像的宽度，也是浏览器的典型宽度）时，我们想把奖杯放在页面中间，并且刚好盖过主内容`<div>`。

因此我们要用top和left属性把奖杯放置在离顶部30像素，离左边365像素的位置。

```
#award {
    position:  absolute;
    top:       30px;
    left:      365px;
}
```

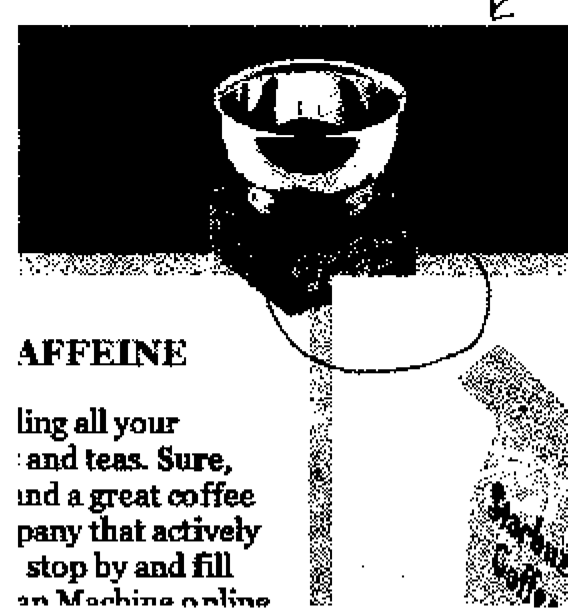
我们把奖杯绝对布置在离顶部30像素，离左边365像素的位置。

把这条CSS添加到“starbuzz.css”文件，保存，并重新加载网页。你会看到奖杯图像魔术般地出现在我们希望的地方。务必调整浏览器看一下奖杯是如何显示的。

一个小问题

当调整浏览器时你有没有注意到一个小问题？取决于你的浏览器，你也许会看到sidebar`<div>`盖过了奖杯图像。到底该怎么办？还记得每个绝对布置的元素有一个描述元素重叠行为的z-index吗？某些浏览器会默认给奖杯元素一个比sidebar`<div>`更低的z-index。你需要做的只是指定奖杯的z-index，给它设置一个比sidebar大的值。

天啊，在一些浏览器上，sidebar会与奖杯图像重叠，并盖住它。

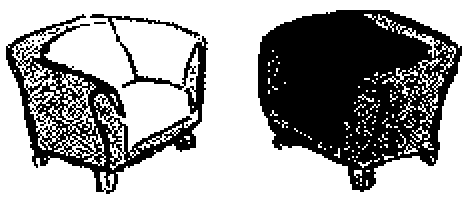



```
#award {
    position:  absolute;
    top:       30px;
    left:      365px;
    z-index:   99;
}
```

我们给奖杯<div>设置一个很大的值，来确保它总是在上面。

继续，把z-index属性添加到“starbuzz.css”文件。保存并重新加载之后，你会发现现在奖杯`<div>`在sidebar上面，重叠的问题解决了。

围炉夜话



今夜话题：Float和Absolute讨论谁更适合于布局

Float

Absolute

Float： Absolute，你有没有注意许多人借助我来做他们的布局？

Absolute： 你确定？你知道，许多页面也用到我。

Float： 好的，每个人都知道我更适合于CSS布局。我使用起来多容易啊。难道你没发现吗？你需要做的只是给你的CSS在合适的位置添加一个小小的float属性，行了！你就有了两栏。

Absolute： 嗯，我好像记得还要一个什么width属性和什么margin属性才能让事情有些头绪……

Float： 细节问题，细节问题！我的观点是，用了我，你不必反复数像素来指出你的内容去哪儿——你只用漂移它，把剩下的交给我就行了。

Absolute： 等一等。我们必须先在XHTML中把整个sidebar移到另一个地方，才能让float开始工作。我不觉得“把剩下的交给你就行了”，那涉及到很多工作。如果是我，至少内容以什么顺序进来是没关系的，我总会正确处理的。

Float： 那么，页脚问题又是怎么回事呢？你总是要用奇怪的方式盖过其他东西，不是吗？如果读者不小心，他们的网页就会有一大块内容正好盖在别的内容上面。至少我可以用clear属性处理这个问题。

Absolute： 嘿，你不也是吗？

Float

但是重要的部分如行内容，流到我周围，是理所应当的。

Absolute

你知道有时人们想把元素正好放置在别的元素上面。用我，你可以把元素放置在任何你想放的地方。这些右和左对你而言都是没有用的。你没给人们那么多的选择，你考虑过这点吗？

Float

你忽略了重点。我更灵活一些，我给了人们很多布置页面的方法。我确信我能做任何你做得到的布局。

Absolute

真的吗？你没有办法做到把奖杯放置得那么酷。

Float

嗯……

Absolute

还是承认了吧！你其实没有我灵活。

Float

那么，或许我做不到你能做的每一件事，不过我想人们用我做基本布局会非常容易，这是人们通常想要的。

Absolute

我不知道，我听说你在旧版的浏览器中有些异常。这对新的网站开发者来说也许会比较棘手。

Float

过去是有那种传闻，但现在大多数现代的浏览器跟我都相处得比较好。并且，现在网站开发者都指名要借助我，正如我开头说的那样。

Absolute

我觉得你还没看到我的全部优点，我在网页中有许多用途。

Float

我从没说过你没用。但是看看那些用了float的设计吧。

Absolute

嗨，别总认为你是完美的。你也许对网页布局有贡献，但你在图形设计中就不一定能做好。

Float

这不是重点，是吗？无论如何，我有float来做清理，行动吧。

Absolute

把这个清理掉，Float。

你该知道的关于绝对布置的另一件事

迄今为止，当你用**left**、**right**、**top**和**bottom**属性指定位置时，这些数字总是跟页面边缘有关，对吗？那么，我们需要稍微改进一下。

当你放置一个元素时，你就把它指定为跟最近放置的祖先元素相关的位置。好的，也许你会说："什么？我除了sibebar什么也没放。我的XHTML中怎么会有已经放置的祖先元素呢？"信不信由你——浏览器创建页面时就把**<html>**元素替你放置好了。

不过，我们再更进一步看一下。假如你想在sibebar中绝对布置一个**<div>**。

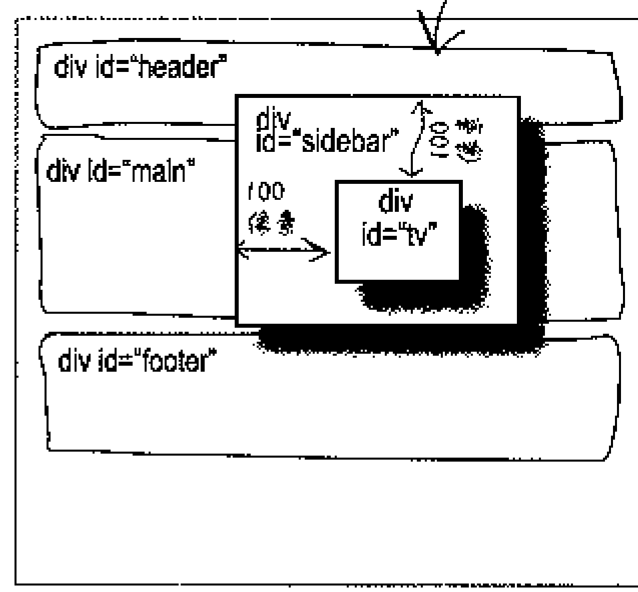
这里是嵌套在sibebar中的一个新的<div>。

```
<div id="sidebar">

    <div id="tv">
        <img src="AsSeenOnTv.gif" alt="As Seen on TV" />
    </div>

    <p class="beanheading">
        ... more XHTML here ...
    </p>
    ...
</div>
```

"tv" <div>的位置与sibebar<div>相对，而不是页面。



如果我们绝对放置"tv"**<div>**，它的最近放置的祖先是sibebar**<div>**。所以它的位置就会跟sibebar相对，而不是页面。

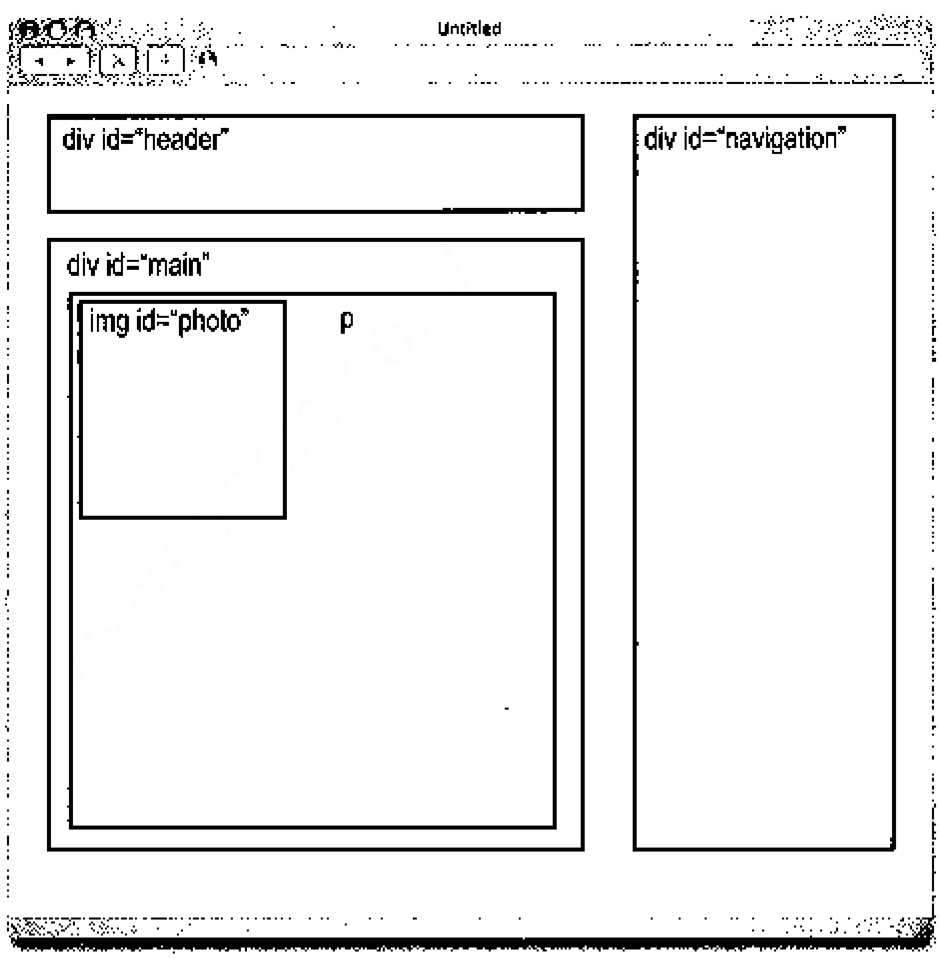
```
#tv {
    position: absolute;
    top:    100px;
    left:   100px;
    width:  100px;
}
```

另一件要知道的事——如果你在下次鸡尾酒会上参加一次关于"最近放置的祖先"的讨论，只须说"最近放置的包含块元素的元素"，这是专家用的术语。

注意！ 如果你把元素放置为与<html>元素相对，bottom属性可能就不会如你所愿。你也许认为"bottom"是网页本身的底部，但是<html>元素实际上把它定义为浏览器窗口的底部。所以，如果你想把一个元素从页面底部绝对布置，而不是浏览器窗口，你需要把元素放进一个延伸到页面底部放置的元素中。实现它的一个方法是把元素放进一个相对布置在页面底部的元素中。以后我们会看到相对布置。

Sharpen your pencil

应该给所有这些有关漂移和布置的知识来一个测试了！看以下的网页，有四个元素，各有一个“id”。你要做的是从右边的CSS规则中找出与这些元素相匹配的id，并把正确的id选择符填上去。核对本章后面的答案。



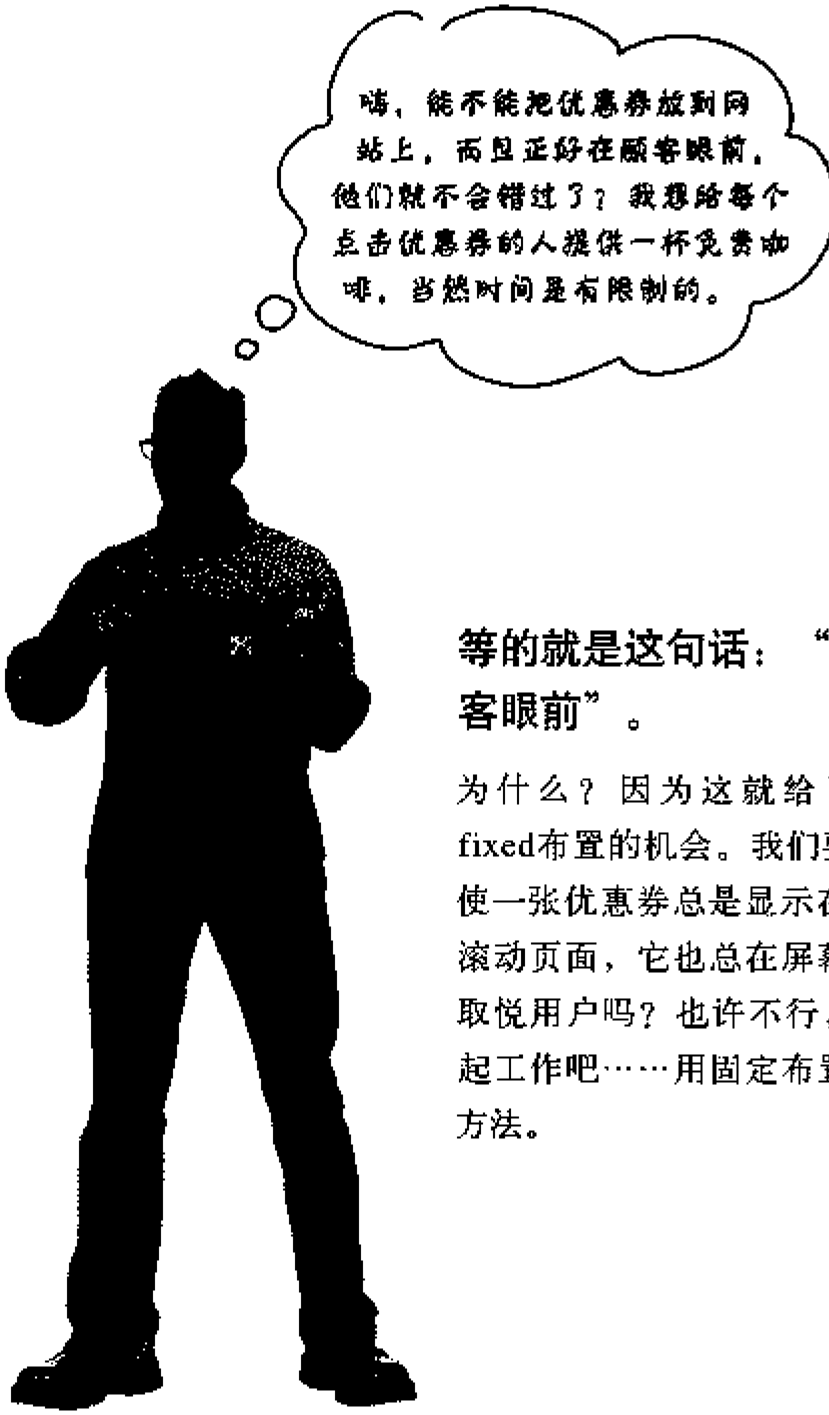
填写选择符完成CSS。

```
____________ {
    margin-top:   140px;
    margin-left:  20px;
    width:        500px;
}

____________ {
    position:  absolute;
    top:       20px;
    left:      550px;
    width:     200px;
}

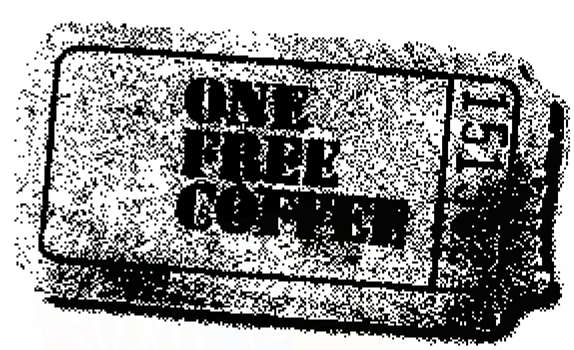
____________ {
    float:  left;
}

____________ {
    position:  absolute;
    top:       20px;
    left:      20px;
    width:     500px;
    height:    100px;
}
```



等的就是这句话：“正好放在顾客眼前”。

为什么？因为这就给了我们一个尝试fixed布置的机会。我们要做的是在网页上使一张优惠券总是显示在屏幕上，即使你滚动页面，它也总在屏幕上。这项技术能取悦用户吗？也许不行，不过先跟我们一起工作吧……用固定布置将是一个有趣的方法。



固定布置如何工作？

跟绝对布置相比，固定布置非常直接。使用固定布置指定元素的位置和你使用绝对布置一样，不过这个位置是相对于浏览器窗口边缘的偏移，而不是页面。这种方法有趣在一旦用固定布置放置了内容，它就呆在你放置的地方，再也不动了，即使你滚动页面。

把浏览器窗口叫做视口来加深朋友和合作者的印象。试试看，很管用，W3C会满意的。

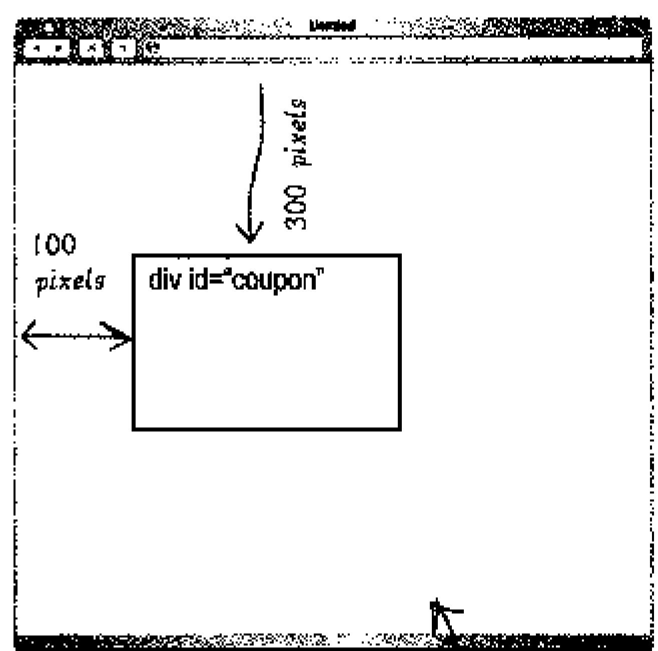
假如你有一个**id**为“coupon”的**<div>**。你可以把这个**<div>**放置在离视口顶部300像素，离左边100像素的一个点上，方法如下：

这是coupon <div>的id选择器。

我们用了固定布置。

```
#coupon {
    position:  fixed;
    top:        300px;
    left:       100px;
}
```

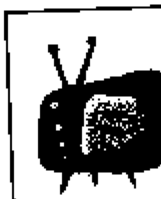
把coupon放置于离顶部300像素，离左边100像素。也可以用right和bottom，和做绝对布置时一样。



这里是元素在视口内放置的地方。

一旦你放置了一个元素，有趣的事就发生了：向各个方向滚动……它根本不动。调整窗口……也不动。拿起你的显示器摇晃……还是不动。好了，最后的说法只是开个玩笑。然而，重点是要知道，固定布置的元素不动，只要显示页面它们就一直在那个位置。

现在你肯定已经想到用固定布置做什么有趣的事了，不过你还有个工作要做。我们要把优惠券放到Starbuzz页面上。



注意！

遗憾的是，IE6.0（以及更早的）版本并不支持固定布置。所以如果你用的是IE，就看不到Starbuzz Coffee网页上的优惠券了。

把优惠券放到页面上

现在我们要把咖啡优惠券放到页面上。我们通过为coupon创建一个**<div>**开始：

这是id为"coupon"的<div>。

里面有优惠券的图像，你会在"chapter12/starbuzz/images"文件夹下找到。

```
<div id="coupon">
    <a href="freecoffee.html" title="Click here to get your free coffee">
        <img src="images/ticket.gif" alt="Starbuzz coupon ticket" />
    </a>
</div>
```

我们把图像用<a>元素包围起来，以便用户可以点击图像来打开新的页面，这样就可以把优惠券打印出来。

把这个**<div>**添加到你的"index.html"文件末尾，就是页脚后面。因为在XHTML中放置的位置只关系到不支持布局的浏览器，并且优惠券还没有重要到一定要放到顶部。

我们编写CSS来放置coupon：

```
#coupon {
    position: fixed;
    top: 300px;
    left: 0px;
}

#coupon img {
    border: none;
}

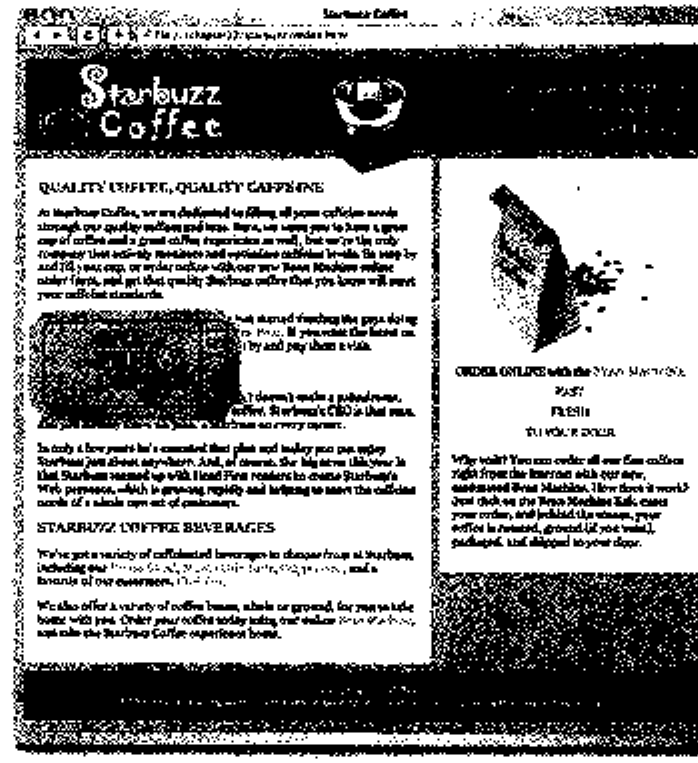
#coupon a:link {
    border: none;
}
#coupon a:visited {
    border: none;
}
```

我们要把优惠券设置成固定布置，离视口顶部300像素，右边靠近视口边缘。所以要指定离左边0像素。

我们还要样式化图像和链接；否则，边框就会出现在图像上，因为图像是可点击的。所以，我们把图像、链接以及已访问的链接的边框都设置为空。

记住CSS中有一条规则表明不用text-decoration，而用边框给链接加下划线。这儿，我们要覆盖在coupon<div>中有关链接的那条规则，表明我们不想在链接上有任何边框。如果你需要让自己回顾一下有关链接的其他规则，回去看看原来的CSS。

把优惠券放到页面上



把新的coupon规则添加到“starbuzz.css”文件，保存后重新加载页面。你可能需要把浏览器调小一些来观察你滚动页面时优惠券仍呆在一个地方。点击优惠券会把你带到“freecoffee.html”页面。

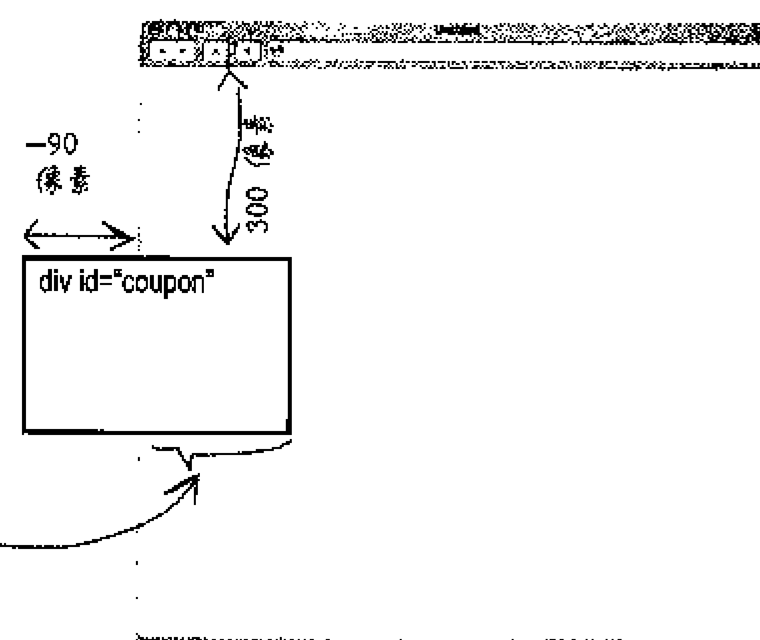
看起来很棒，但如果优惠券偏向左边的话可能会显得更时髦一些，看起来就好像它是从视口边上出来的一样。现在，可以用我们的图片编辑软件把图像的左边削去来创造这种效果。或者，我们可以只用负的偏移来把图像的左边放进视口边缘的左侧。对，你可以这么做。

使用负的left属性值

定义一个负的属性值跟定义正的一样：只需在数字前加一个负号。像这样：

```
#coupon {
    position: fixed;
    top: 300px;
    left: -90px;
}
```

定义为–90像素，就是告诉浏览器把图像放到视口边缘的左侧90像素处。

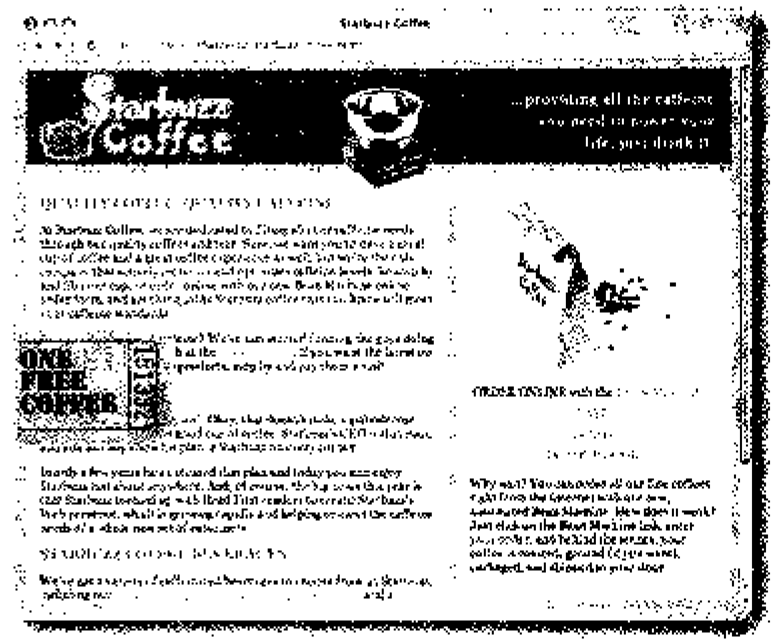


浏览器会很高兴地为你把图像放到视口左边，这样就只能看到保留在屏幕上的那部分图像了。

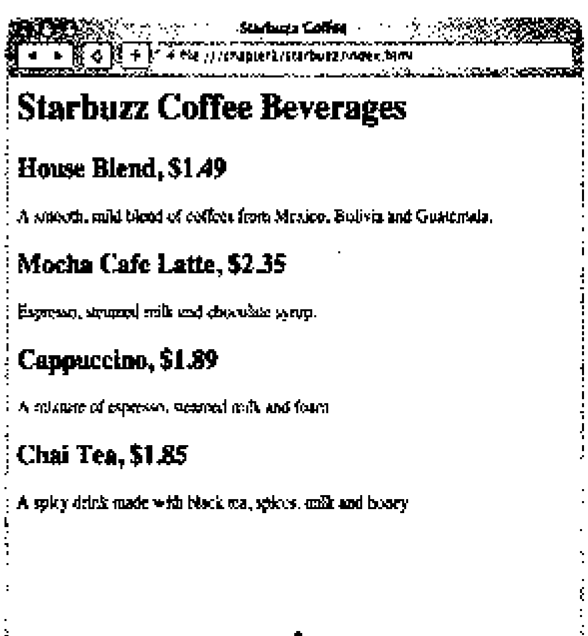
正的、负的属性值测试

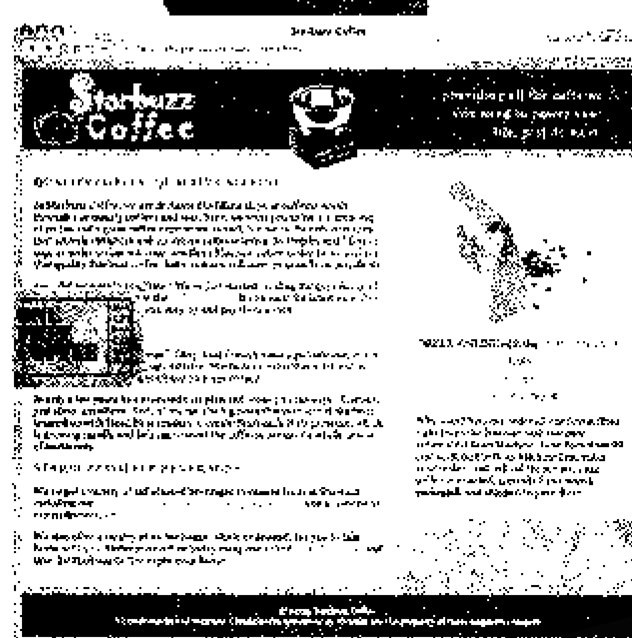
务必添加负的left属性值，保存并重新加载页面。看起来很漂亮吧？祝贺你，你刚完成了你的第一个CSS特效。小心George Lucas！

要记住，用固定布置掩盖内容相对用户来说并不是最友好的，不过很有趣。









哇，简直是天壤之别！

了解相对布置

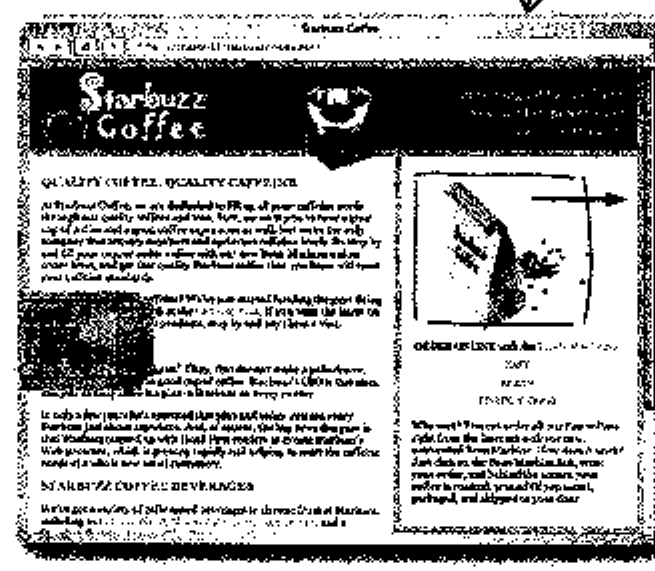
这是最后一种布置：相对布置。说实话，它也是最孤单的布置，因为你会发现没有多少人在他们的设计中用到它。不过，每天都有新的设计出现，所以当你看到相对布置时，你会想知道它是如何工作的以及它是做什么的。

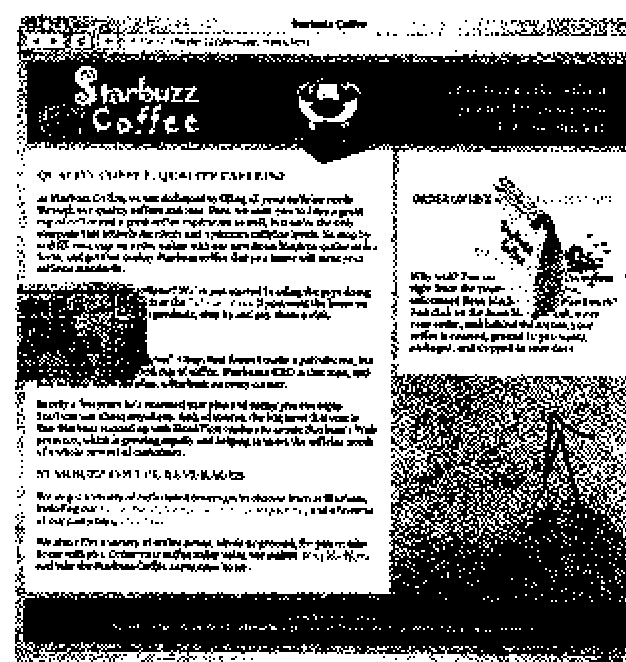
不像绝对布置和固定布置，相对布置的元素仍然是页面流中的一部分，不过在最后一刻，就在元素被显示之前，浏览器偏移它的位置。我们在Starbuzz页面的咖啡袋上试试，看看它是如何工作的。我们要把咖啡袋移到旁边，那么那些从袋子中散出的咖啡豆看上去就会像正从袋中散出一样。

现在我们可以绝对布置咖啡袋，但是如果我们这么做了，就必须找一种方法在页面上留出一定的空间，因为绝对布置是把文档从流中完全移走。

这就要由相对布置来解决了。我们可以让元素仍然在流中，把它的空间留出来，然后把它偏移到它实际要显示的地方。让我们试试吧！

我们想把Starbuzz咖啡袋向右移大约100像素。





用绝对布置来做的话，咖啡袋移动了，不过因为它不在流中了，页面其他的部分就会移到它下面，被它覆盖。

这里是一条选择图像的新规则。在这里我们用子孙选择符来选择只在beanheading里的图像。

注意图像是内联元素，但没关系，你可以用任何一种布置技术，其实在内联元素上用漂移也行。

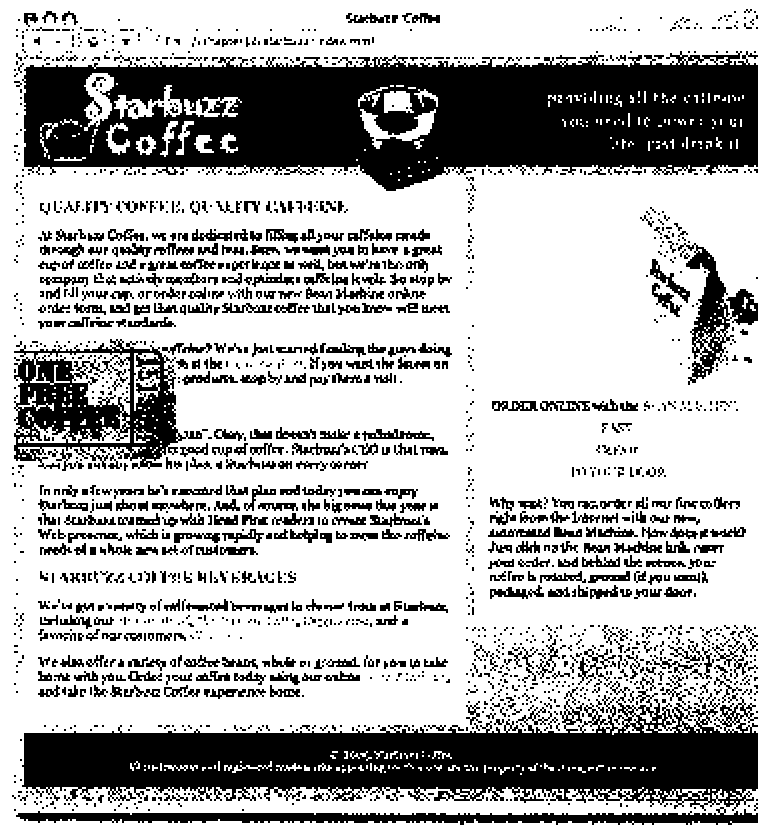
```
.beanheading img {
    position: relative;
    left: 120px;
}
```

然后我们定义了一个相对位置，以及我们希望图像偏移的值。偏移与它在流中放置的位置相对应。

所以这里我们指定图像应该显示在相对于文档流位置的左边120像素。在指定偏移的时候，你也可以用right、top和bottom。

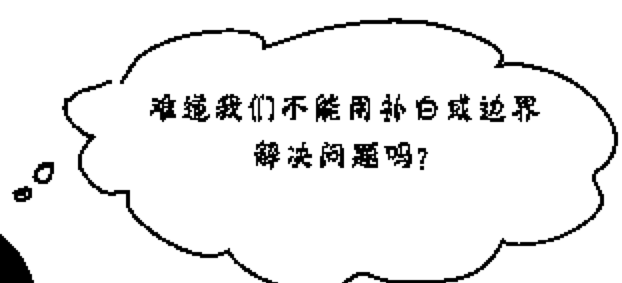
添加这个规则到你的CSS，然后保存并重新加载。

测试



重新加载Starbuzz页面后，你会看到咖啡袋移到了sidebar的右边。有趣的是图像的一部分因为扩展超过sibebar到了边界，消失在页面的边缘。这又是为什么呢？好的，如你所见，浏览器先把相对元素流到页面上，然后才把它偏移到要显示的地方。所以元素在页面上还是占据了同样大的地方，只是画在了不同的位置。相对布置稍微有些像静态布置，又稍微加了点绝对布置。不过，不像绝对布置，相对布置被定义为相对于元素实际位置的偏移，而不是与最近包含的块绝对地平行。

所以，这改进页面了吗？不敢肯定，不过很有趣（你也许想在CEO看页面之前，把相对布置去掉）。





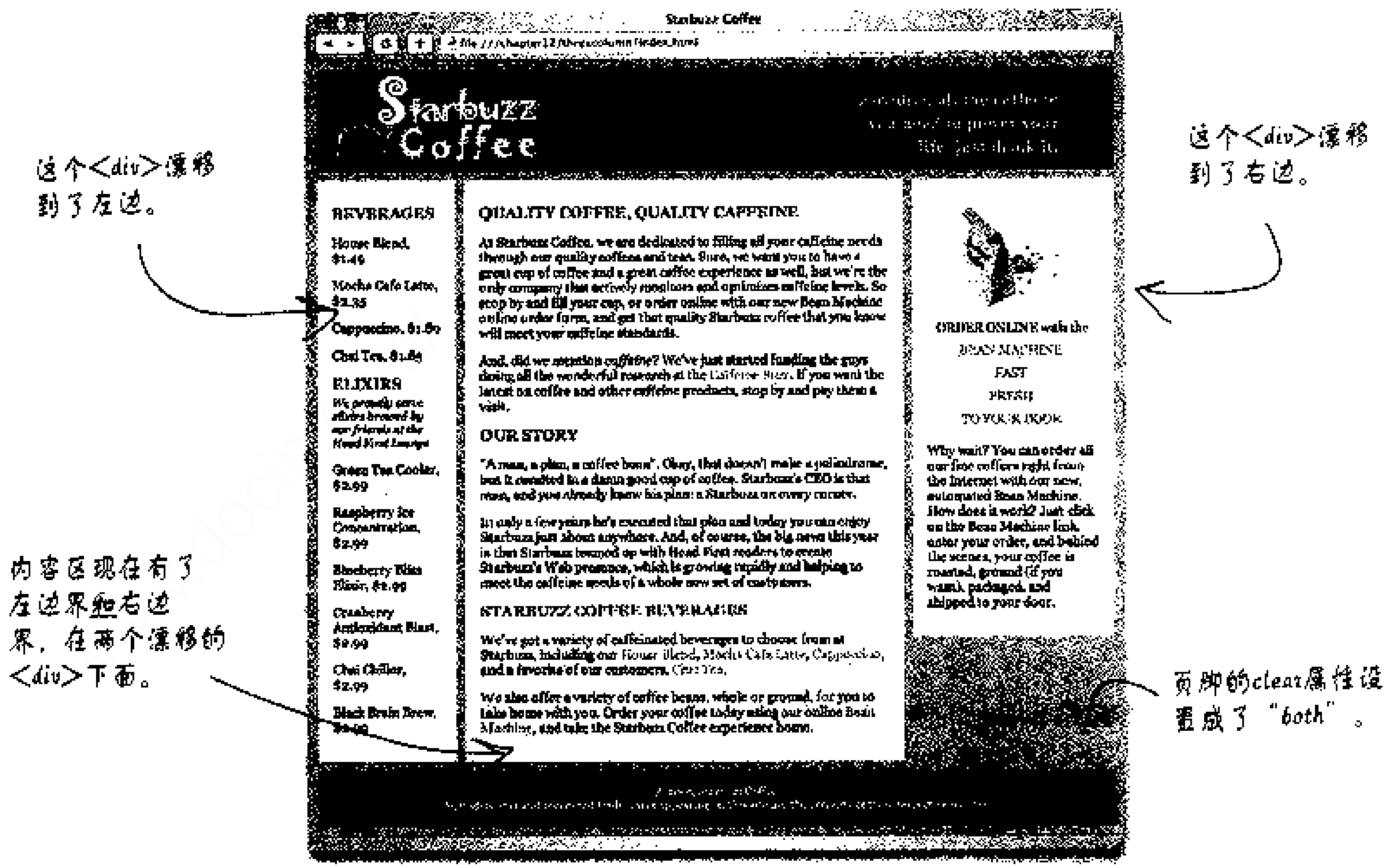
不能。

无论你如何调整补白和边界，你还是不能把一个图像放到它所在盒子的外边。为什么要用那么难的方法做呢？用两行CSS就可以达到更好的效果。你可以用相对布置来让一个元素在流中越过自己的盒子，以便很好地显示，用补白和边界就做不到这点。

做成三栏甚至更多栏……

我们在这章中一直着眼于两栏布局，真正的目的就是学习**float**和**clear**属性，以及CSS提供的多种形式的布局方法。现在你已经有了这些基础，可以考虑做三栏布局，或者你希望的任何其他布局。好了，这章结束了。

不过，等等！在你结束之前，我们想一下如何做三栏布局（如果你想试试，在“chapter12/threecolumn”文件夹下就能找到）。



这个设计是用你已经理解的技术做的。去探索你在这儿没做过的东西吧，看看别人如何用CSS创建有趣的设计确实会对你很有帮助，我们鼓励你走出去四处看看。你可以参考我们最喜欢的有关CSS设计的在线资源，网址是：

http://headfirstlabs.com/books/hfhtml/chapter12/cssdesign.html

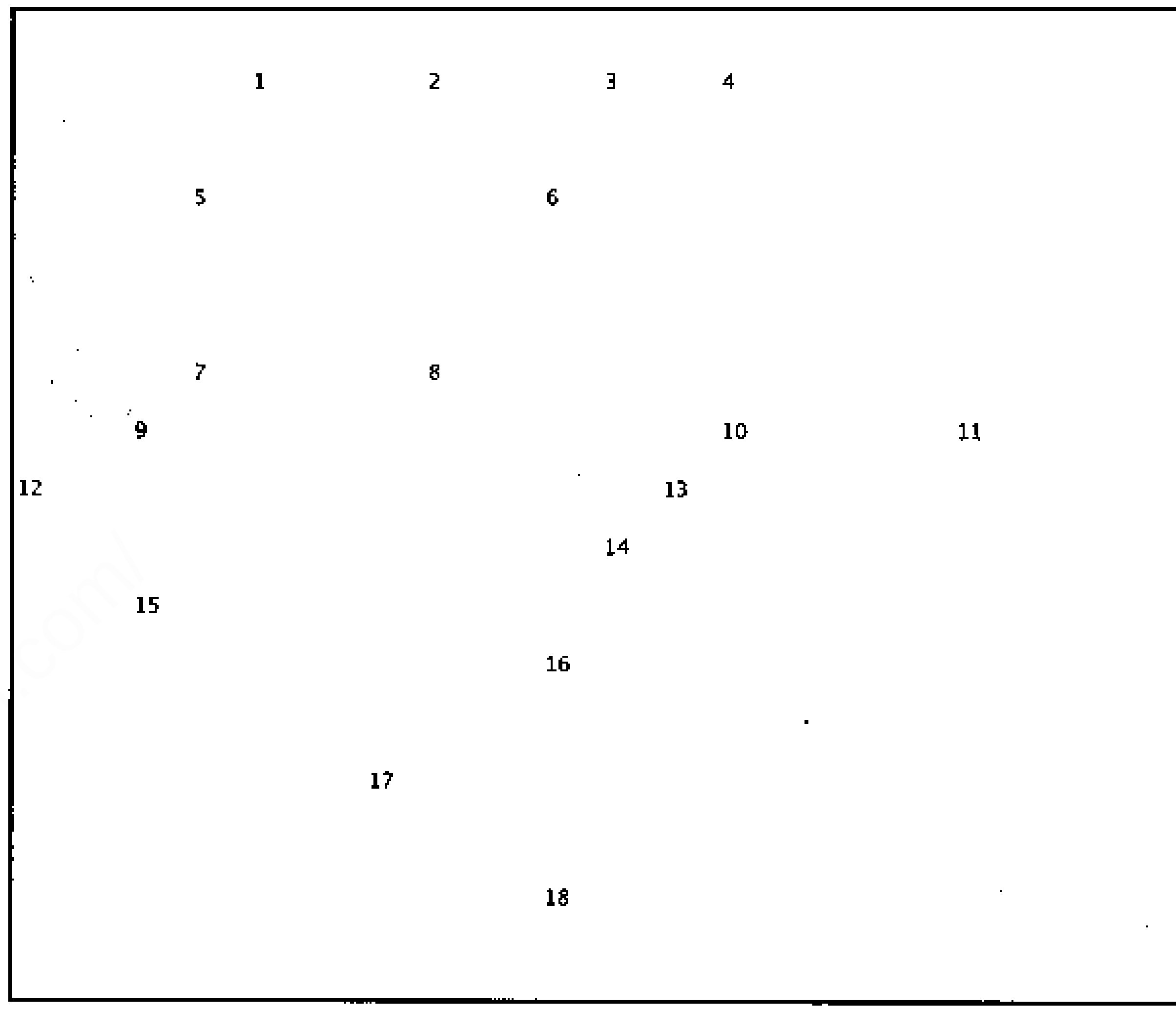
要点

- 浏览器用流在页面上放置元素。
- 块元素从开始流到结尾，元素之间有换行。每个块元素默认占据浏览器窗口的整个宽度。
- 内联元素在块元素中从左上方流到右下方。如果需要多于一行的空间，浏览器就新创建一行，在垂直方向上扩展包含块元素以包含内联元素。
- 普通页面流中的两个并列放置的块元素，上边元素的下边界和下边元素的上边界会重叠，它们之间的边界就是较大边界的大小，或者如果大小相等就是一个边界的大小。
- 漂移元素被移出了正常的流，放到左边或右边。
- 漂移元素放置在块元素的上面，不影响它们的流。不过，行内容考虑漂移元素的边界，流到它的周围。
- clear属性是用来指定一个块元素的左边或右边（或两边）不能有漂移元素。一个设置了clear属性的块元素会一直向下移直到它边上没有块元素。
- 漂移元素必须设置一个明确的宽度值，不能是默认的。
- 流动布局是：当你扩大浏览器窗口时，页面内容也随之扩大来填满页面。
- 冻结布局是：内容的宽度是固定的，不随浏览器窗口的变化而扩大或缩小。这有一个优点是你可以更好地控制你的设计，但代价是不能有效地利用浏览器宽度。
- 凝胶物布局是：内容区的宽度是固定的，但边界随着浏览器窗口的变化扩大或缩小。凝胶物布局通常用来把内容放在页面的中间。它跟冻结的布局有一样的优点，但总是更引人注目一点。
- 可以把position属性设置为四个值：static、absolute、fixed和relative。
- static是默认布局，把元素放在页面的正常流中。
- 用绝对布置可以把元素放在页面的任何位置。默认状态下，绝对布置相对于页面边放置元素。
- 如果把一个绝对布置嵌入另一个布置过的元素中，那么它的位置就是相对于布置过的那个包含它的元素。
- top、right、bottom和left属性用来放置绝对、固定和相对布置的元素。
- 绝对布置的元素可以用z-index属性互相层叠放置。较大的z-index值表明它在上层（在屏幕上更靠近你）。
- 固定布置的元素总是以相对于浏览器窗口的位置放置，页面滚动时不移动。页面中其他内容在这些元素下面滚动。
- 相对布置的元素先和正常流一样流到页面上，然后按照指定的数字偏移，在它们应该正常显示的地方留出了空间。
- 用相对布置时，left、right、top、bottom涉及到与元素在正常流中的位置相对的偏移量。
- 通常使用漂移元素或绝对布置的元素可以完成同样的设计。
- float提供了一种形成两栏布局的灵活的解决方法，元素可以把漂移元素从它们身边清理掉，绝对布置做不到这点。



XHTML填字游戏

这一章是turbo负责的，有很多东西要学。通过做这个填字游戏加深对知识的理解。答案在本章末尾。



横排提示：

5. 流动的和冻结的之间的状态。
6. 相对于视口的一种布置。
7. 当你相邻放置两个内联元素时，它们的边界不____。
12. 一般来说____是做成分栏布局更好的一种技术，因为你可以用clear。
13. 内联元素从____上开始流。
15. 一种特殊的内联元素，当页面展开时一起组成盒子。
16. 绝对布置相对于布置过的____块。
17. 这种边界用于优惠券来造就特殊的效果。
18. 通常用来指出一个要布置的元素。

竖排提示：

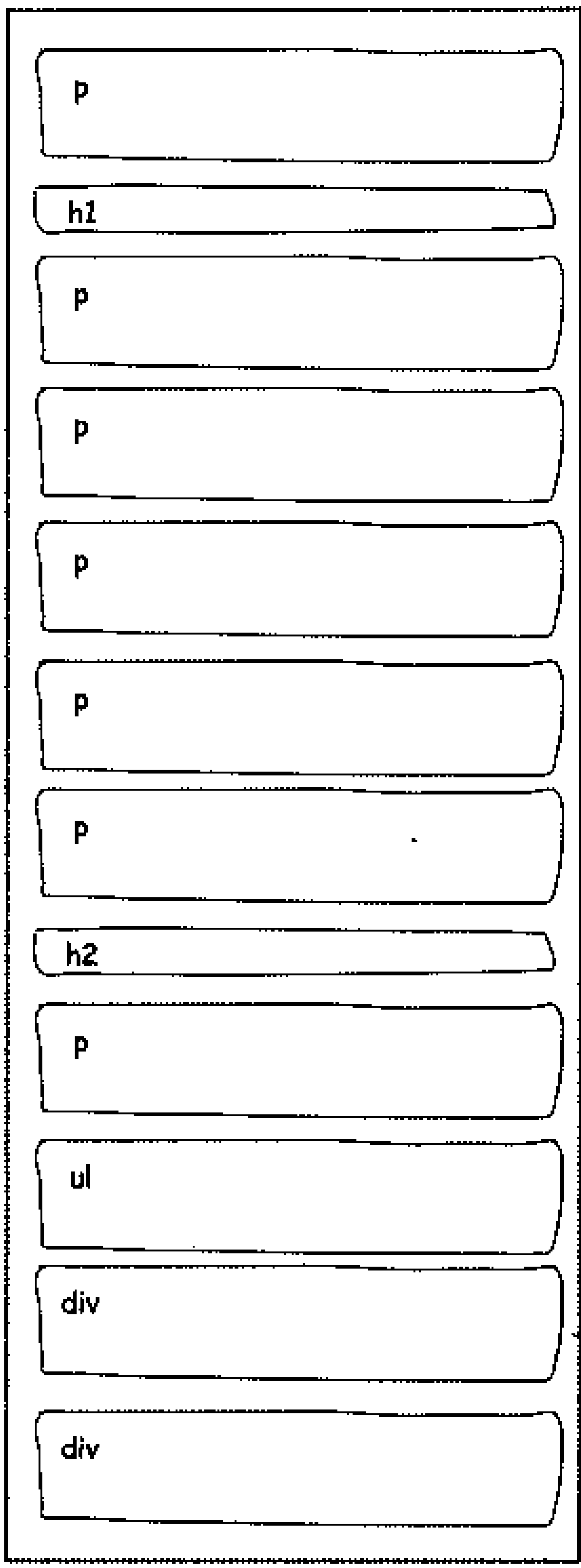
1. 浏览器的另一个名字。
2. 浏览器用来在网页上放置static元素的方法。
3. 描述布置过的元素的分层行为的属性。
4. 用来解决覆盖页脚问题的属性。
8. 用这种布置，可以把位置指定为相对于包含块的边缘。
9. 块元素从开头流到____ 。
10. 一种把元素保留在流中的布置方法。
11. 当盒子被并列放置时，这些会重叠。
12. 把元素从流中移走，设置到一边。
14. 行内容流到____元素周围。



扮演浏览器



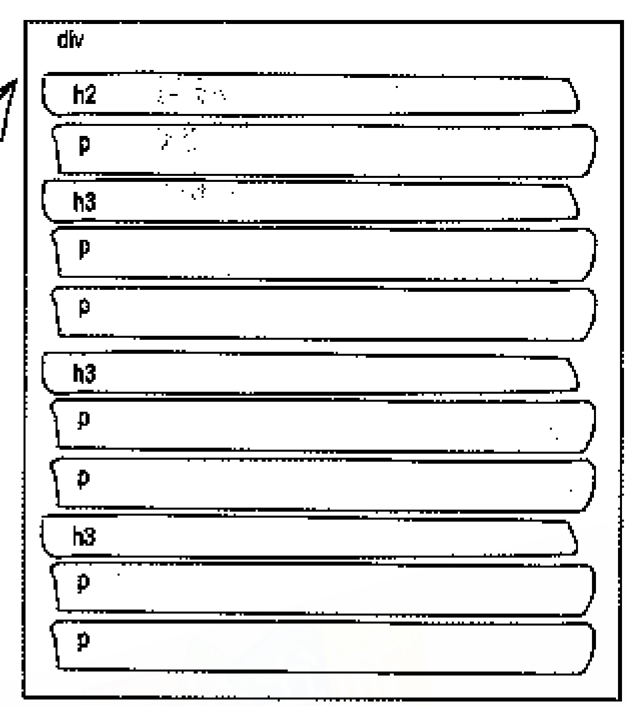
打开"lounge.html"文件并找到所有块元素。把每个元素流到左边的页面中。注意力集中在直接嵌套在body元素中的块元素上。也可以忽略CSS中的"float"属性，因为你还不知道它是什么。答案如下。



在"lounge.html"文件中的每个块元素从开头流到结尾，块元素之间有换行。

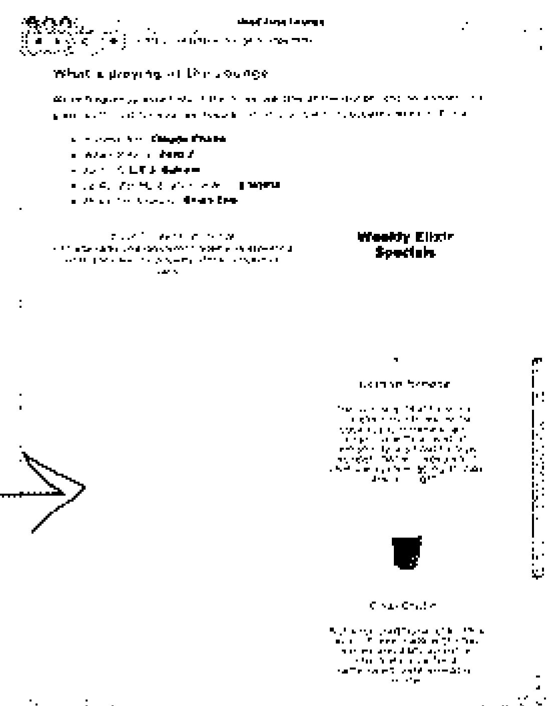
这三个元素中嵌套了块元素。

我们不要求你这么做，不过如果你想多做一些，这里是它们流入的方法。



练习答案

把饮料<div>移回原来音乐推荐下面的那个位置，保存并重新加载页面。现在那个元素漂移到哪儿了？你应该看到音乐推荐底下的饮料了吧。



<div>漂移到了右边，恰好在音乐推荐下，XHTML中剩下的部分漂移到它周围（只有页脚）。

Sharpen your pencil

答案

我们要做的是在主内容部分设置一个与sidebar一样宽的右边界。但是sidebar有多大呢？希望你还没忘记上一章学过的内容。以下是你计算sidebar宽度要用到的所有信息。答案如下：

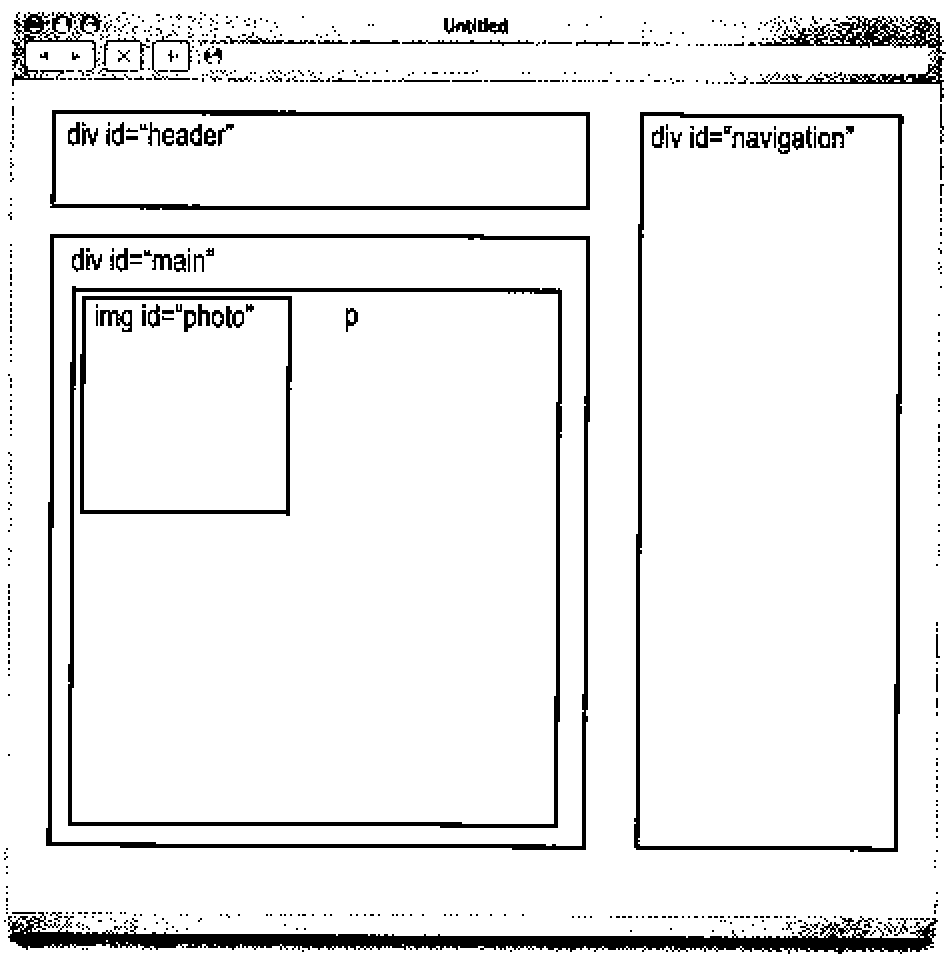
```
#sidebar {
        background: #efe5d0 url(images/background.gif) bottom right;
        font-size:  105%;
        padding:    15px;
        margin:     0px 10px 10px 10px;
        width:      280px;
        float:      right;
}
```

15 + 15 + 280 + 0 + 0 + 10 + 10 = 330

左填充　右填充　内容区　左边框　右边框　右边界　左边界

Sharpen your pencil 答案

应该给所有这些有关漂移和布置的知识来一个测试了！看以下的网页，有四个元素，各有一个“id”。你要做的是从右边的CSS规则中找出跟这些元素相匹配的id，并把正确的id选择符填进去。答案如下，你全做对了吗？

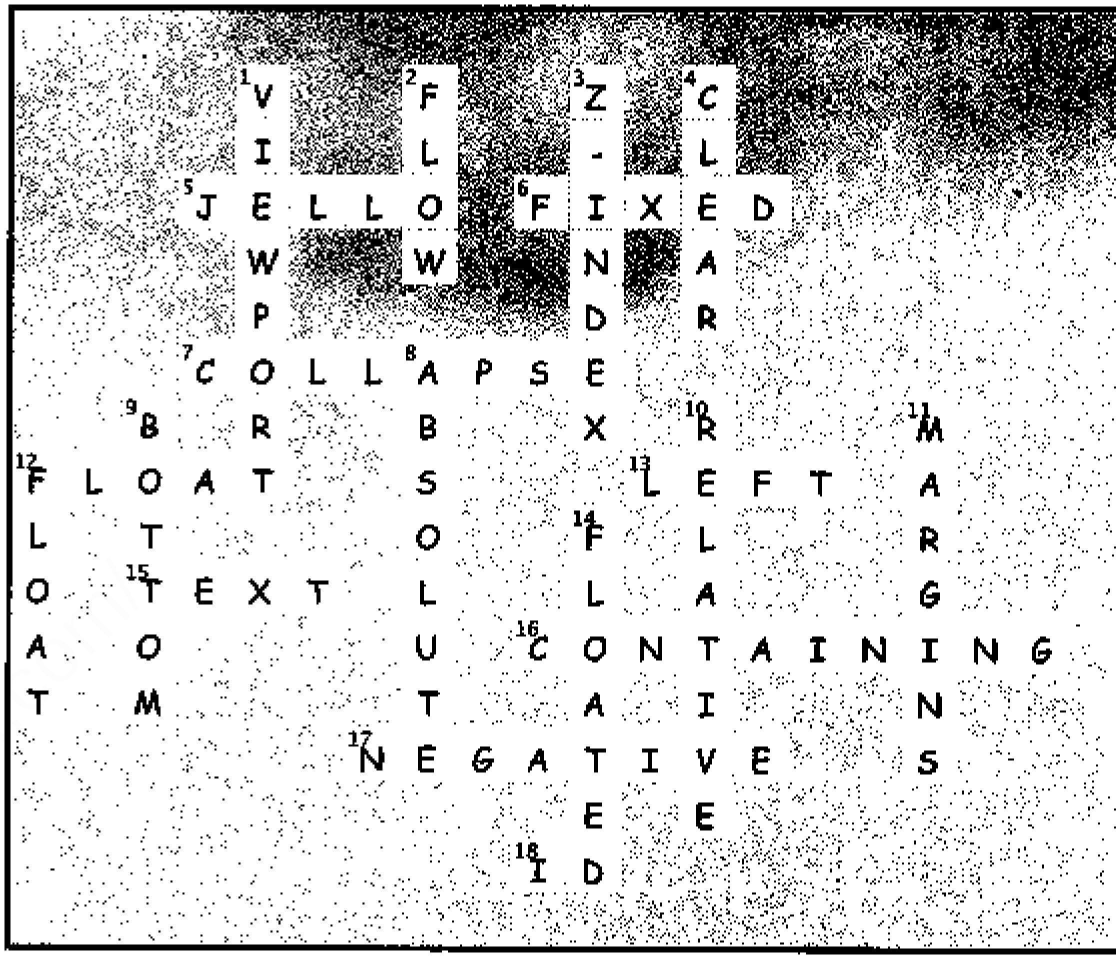


填写选择符完成CSS。

```
#main {
    margin-top:   140px;
    margin-left:  20px;
    width:        500px;
}
#navigation {
    position: absolute;
    top:      20px;
    left:     550px;
    width:    200px;
}

#photo {
    float: left;
}
#header {
    position: absolute;
    top:      20px;
    left:     20px;
    width:    500px;
    height:   100px;
}
```

解答



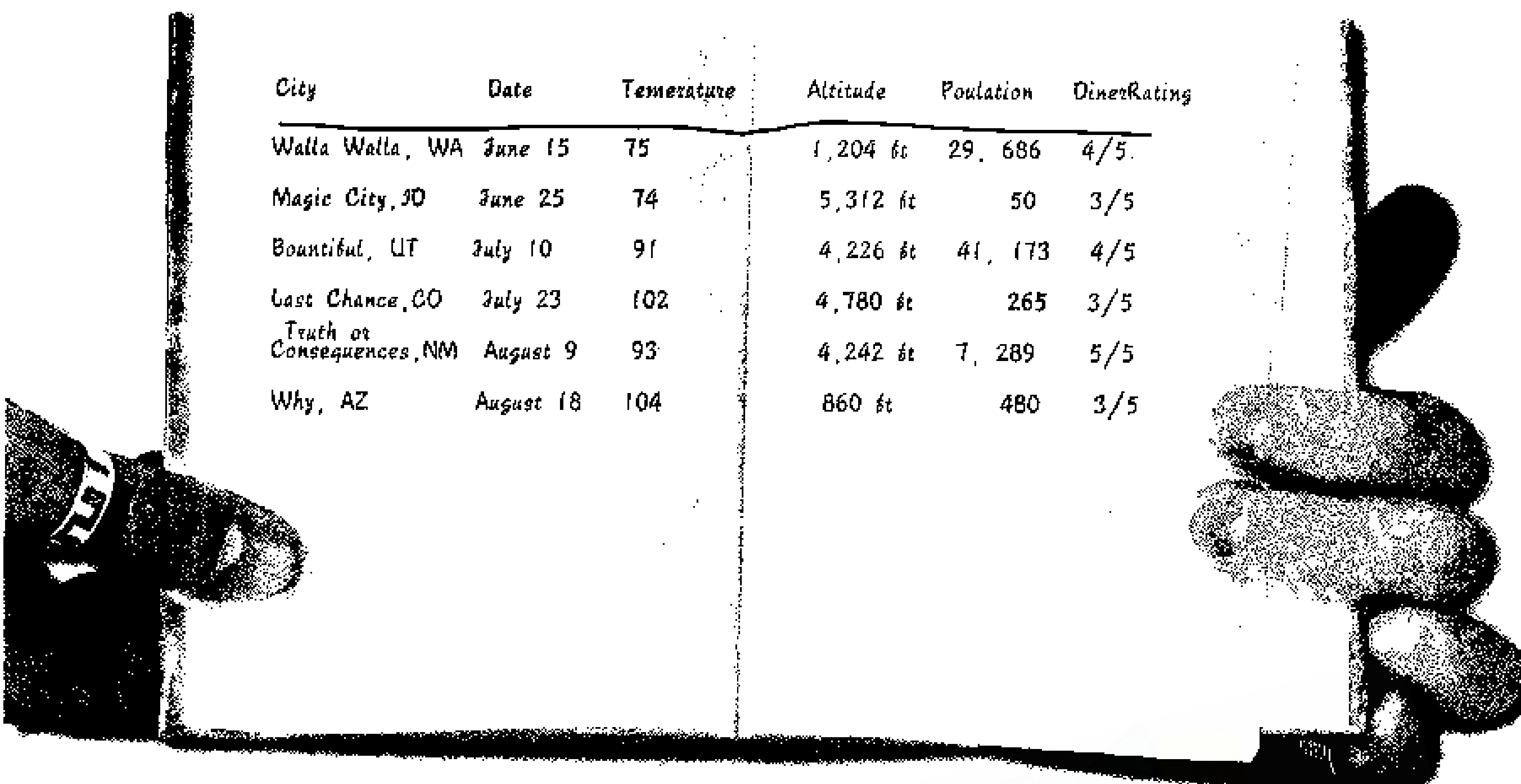
[1]V [2]F [3]Z [4]C
I L - L
[5]J E L L O [6]F I X E D
W W N A
P D R
[7]C O L L [8]A P S E
[9]B R B X [10]R [11]M
[12]F L O A T S [13]L E F T A
L T O [14]F L R
O [15]T E X T L L A G
A O U [16]C O N T A I N I N G
T M T A I N
[17]N E G A T I V E S
E E
[18]I D

开始制作表格



如果像表格那样行走和交谈……生活中，我们常常要处理一些令人头痛的表格式数据。不管是创建一个描述你公司去年的存货清单的网页，还是制作一个布娃娃收藏的编目表（别担心，我们不会说出去），你都需要用XHTML来实现；但如何实现呢？在这一章中我们将解读表格的秘密让你将个人数据正确地存入XHTML表格。此外，如果你现在就行动，作为特殊的奖励，我们将抛出指南引导你设计XHTML列表。别犹豫，行动吧！





City	Date	Temerature	Altitude	Poulation	DinerRating
Walla Walla, WA	June 15	75	1,204 ft	29,686	4/5
Magic City, ID	June 25	74	5,312 ft	50	3/5
Bountiful, UT	July 10	91	4,226 ft	41,173	4/5
Last Chance, CO	July 23	102	4,780 ft	265	3/5
Truth or Consequences, NM	August 9	93	4,242 ft	7,289	5/5
Why, AZ	August 18	104	860 ft	480	3/5

如何用XHTML创建表格？

Tony是对的。你确实还没有找到一个运用XHTML的好方法来描述他的表格，至少现在还没有。那么，你可能已经想到一个方法，即用CSS和**<div>**来创建表格，不过，XHTML中的**<table>**元素能够满足制作表格的所有需求。所以，在学习**<table>**之前，我们先来了解一下表格：

我们有列……

这一行包含的表头。

和行……

City	Date	Temp	Altitude	Population	Diner Rating
Walla Walla, WA	June 15th	75°	1,204 ft	29,686	4/5
Magic City, ID	June 25th	74°	5,312 ft	50	3/5
Bountiful, UT	July 10th	91°	4,226 ft	41,173	4/5
Last Chance, CO	July 23rd	102°	4,780 ft	265	3/5
Truth or Consequences, NM	August 9th	93°	4,242 ft	7,289	5/5
Why, AZ	August 18th	104°	860 ft	480	3/5

我们把每一小块数据叫做单元格，或者有时叫做表格数据。

BRAIN POWER

如果让你来设计XHTML，你会如何制定那些定义表格的一个或多个元素？表格包括表头、行、列和实际表格数据。

怎样用XHTML创建一个表格

在我们引进Tony的网站并对它做一些修改之前，我们先在一个单独的XHTML文件中测试表格。在chapter13/journal文件夹下,打开“table.html”XHTML文件找到代码中的表头和表格的前两行。检查一下：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">
<head>
   <meta http-equiv="content-type" content="text/html; charset=ISO-8859-1" />
   <style type="text/css">
        td, th {border: 1px solid black;}
   </style>
   <title>Testing Tony's Travels</title>
</head>
<body>
   <table>
      <tr>
         <th>City</th>
         <th>Date</th>
         <th>Temperature</th>
         <th>Altitude</th>
         <th>Population</th>
         <th>Diner Rating</th>
      </tr>
      <tr>
         <td>Walla Walla, WA</td>
         <td>June 15th</td>
         <td>75</td>
         <td>1,204 ft</td>
         <td>29,686</td>
         <td>4/5</td>
      </tr>
      <tr>
         <td>Magic City, ID</td>
         <td>June 25th</td>
         <td>74</td>
         <td>5,312 ft</td>
         <td>50</td>
         <td>3/5</td>
      </tr>
   </table>
</body>
</html>
```

一小块CSS程序，我们可以在浏览器中看到表格的结构。现在还不必担心这个。

用一个<table>标记，开始一个表格。

这是第一行，以一个<tr>标记开始。

每个<th>元素构成一列的表格表头。

注意：表格表头是逐个罗列的，看上去这些元素构成了一行，实际上我们是定义了整个表头行。回过头看看Tony的列表，他是如何绘制这些表头的。

每个<tr>元素构成一个表格行。

这是第二行的开始，也就是Walla Walla城市。

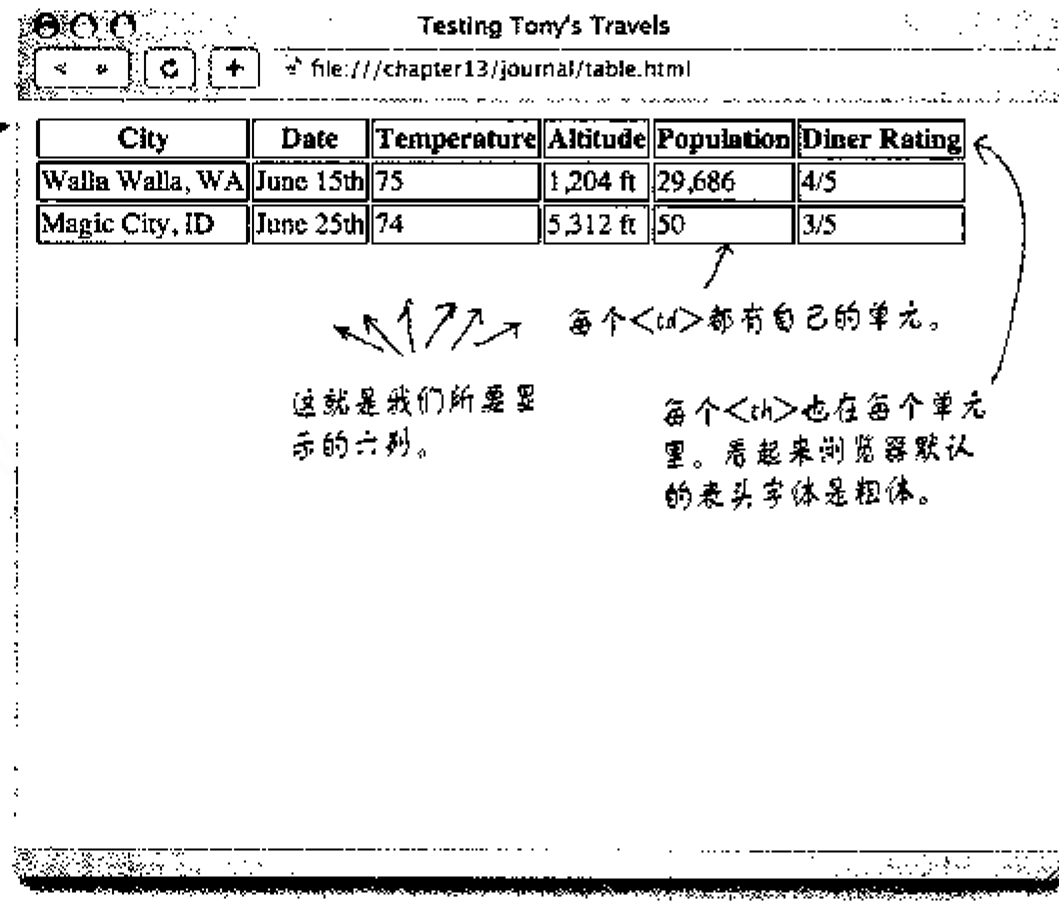
每个<td>元素占据表格的一个单元，每个单元都是单独的一列。

所有的<td>标记组成一行。

这是第三行。每个<td>元素构成一个表格数据。

浏览器生成了什么?

让我们来看看浏览器是如何显示这个XHTML表格的。注意：这不是一个外表好看的表格，只是它看起来像一个表格。而我们担心的是它看上去太简短了；现在让我们确信你已经掌握了表格的基础知识。

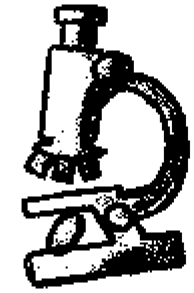


City	Date	Temperature	Altitude	Population	Diner Rating
Walla Walla, WA	June 15th	75	1,204 ft	29,686	4/5
Magic City, ID	June 25th	74	5,312 ft	50	3/5



首先，把前一页的程序键入名为“Testing Tony's Travels”的文件中，键入的过程虽然有点单调，却能使你了解<table>、<tr>、<th>、<td>等标签的结构。完成后，快速测试一下，接着添加表格中的剩余项目，再测试。

表格剖析



你已经知道了创建单一表格的四个元素：<table>，<tr>，<th>，<td>。让我们进一步了解每个元素，特别是它们在表格中所起的作用。

<table>标记是整个表格的开始标记。如果你想创建一个表格，那么以它开始。

<th>元素包含一个单元格，它必须位于表格行中。

<th>Date</th>

用</tr>标记结束一个表格行。

<table>

City	Date	Temp	Altitude	Population	Diner Rating
Walla Walla, WA	June 15th	75°	1,204 ft	29,686	4/5
Magic City, ID	June 25th	74°	5,312 ft	50	3/5
Bountiful, UT	July 10th	91°	4,226 ft	41,173	4/5
Last Chance, CO	July 23rd	102°	4,780 ft	265	3/5
Truth or Consequences, NM	August 9th	93°	4,242 ft	7,289	5/5
Why, AZ	August 18th	104°	860 ft	480	3/5

（每一行左侧标注 **<tr>**，右侧标注 **</tr>**）

每个<tr>元素定义一个表格行。所有的表格数据都是嵌入在每个<tr>元素中的。

<td>August 9th</td>

<td>元素包含一个表格中的数据单元。它必须放在一个表格行中。

</table>

以</table>标记结束这个表格。

there are no Dumb Questions

问： 为什么表格中没有列元素？它看起来也很重要。

答： XHTML语言设计者用行元素定义表格，而不是列元素。不过，我们注意到在指定行中的每一个<td>元素的同时，实际上也隐式地定义了每一列。

问： 如果一行中元素不够将会出现什么情况？换句话说就是，表格项少于列的个数怎么办？

答： 最简单的办法就是不去理睬，保留内容为空的单元格。换句话说，你书写了一个空的<td></td>语句。如果你省去了单元格的数据项，表格排列就不整齐。所以，即使数据单元格是空的，还是要罗列出所有的数据项。

问： 如果我想将表头放在表格的左边而不是上边，我该如何去做呢？

答： 你的想法是可以实现的。你只需要将表格表头元素分别放入每行来代替原先的第一行。把每个<th>元素放在每行的开头，那么所有的表头就在第一列了。

问： 我的朋友给我展示了一种技巧，完全只用表格来完成页面的版面设计，他甚至都没有使用CSS。

答： 直接进入CSS。不学会CSS，你是找不到工作的。

在CSS出现之前，使用表格来完成版面设计是非常普遍的。坦率地说，当时没有更好的方法来做一些复杂的版面设计。不过，在今天，使用表格做版面设计却是一个糟糕的方法。如果完全只使用表格，既困难又难以维护。告诉你的朋友，他的那种技巧已经过时了。他应该学习正确的方法，也就是应该使用XHTML和CSS做版面设计。

问： 表格不就是外观设计吗？它与结构有什么关联？

答： 也不全是。你可以用表格指定表格式数据之间的关系。我们一般使用CSS来改变表格的外观。

表格提供了一种在HTML中定义表格式数据的方法。

表格由行中的数据单元组成。

行中隐式地定义了列。

表格中列的个数，取决于一行中数据单元格的个数。

总的来说，表格的作用不在于外观，外观是CSS的工作。

扮演浏览器

左边是一段表格的XHTML语句。你的任务就是扮演浏览器来显示表格。做完后，参考这一章后面的答案，查看你是否做对了。

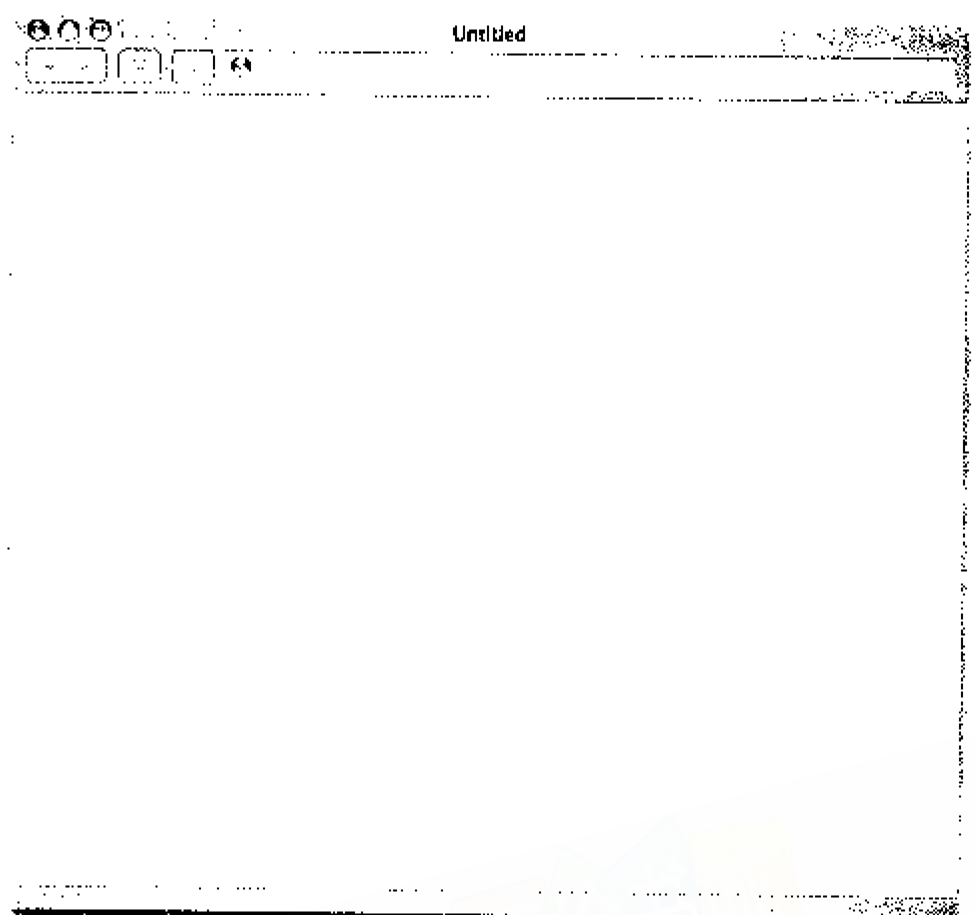


```
<table><tr><th>Artist</th>
<th>Album</th></tr><tr>
<td>Enigma</td><td>Le Roi Est Mort,
Vive Le Roi!</td></tr> <tr><td>LTJ
Bukem</td>
<td>Progression Sessions 6</td>
</tr><tr>
<td>Timo Maas</td>
<td>Pictures</td></tr></table>
```

这就是表格XHTML。

啊！有人需要学习书写XHTML的格式了。

在这儿画出表格。



添加标题和摘要

要优化你的表格，有两件事情你要立刻去做：一是添加一个标题，二是添加一个摘要。

```
<table summary="This table holds data about the
cities I visited on my travels.  I've included the date I
was in each city, the temperature when I was there, and
altitude and population of each city.  I've also included
a rating of the diners where I had lunch, on a scale
from 1 to 5.">
      <caption>
            The cities I visited on my
            Segway'n USA travels
      </caption>
      <tr>
         <th>City</th>
         <th>Date</th>
         <th>Temperature</th>
         <th>Altitude</th>
         <th>Population</th>
         <th>Diner Rating</th>
      </tr>
      <tr>
         <td>Walla Walla, WA</td>
         <td>June 15th</td>
         <td>75</td>
         <td>1,204 ft</td>
         <td>29,686</td>
         <td>4/5</td>
      </tr>
      <tr>
         <td>Magic City, ID</td>
         <td>June 25th</td>
         <td>74</td>
         <td>5,312 ft</td>
         <td>50</td>
         <td>3/5</td>
      </tr>
      .
      .
      .
   </table>
```

摘要不会在你的网页中显示出来。它的作用纯粹只是增加表格的可读性，起一小块文本的作用，还能使屏幕读取器能够更好地读取用户所描述的表格。

标题，就是浏览器中的标题。默认状态下，多数浏览器将标题设置在表格上方。

如果你不喜欢标题的默认位置，你可以用CSS重新定位标题（我们一会儿就试一下），尽管有些浏览器并不完全支持重置标题的默认格式。

表格剩余的行就放在这个地方。

测试……开始考虑表格的样式

把摘要和标题添加到你的表格中，保存并加载页面。

你看不到摘要，它不会显示出来。屏幕读取器首先读入摘要，能够为视力受损者提供更多的表格信息。

标题位于表格的上面。不过，如果把它放在表格的下面也许看起来更好一点。

Tony's Table

file:///chapter13/journal/table.html

The cities I visited on my Segway'n USA travels

City	Date	Temperature	Altitude	Population	Diner Rating
Walla Walla, WA	June 15th	75	1,204 ft	29,686	4/5
Magic City, ID	June 25th	74	5,312 ft	50	3/5
Bountiful, UT	July 10th	91	4,226 ft	41,173	4/5
Last Chance, CO	July 23rd	102	4,780 ft	265	3/5
Truth or Consequences, NM	August 9th	93	4,242 ft	7,289	5/5
Why, AZ	August 18th	104	860 ft	480	3/5

我们需要在表格数据单元格之间添加补白，使得单元格数据更易阅读。

要是加些橙色与Tony的网站相匹配，整体效果就更好了。

边框线好像浓了一点。我们可以在单元格中使用淡些的边框线（尽管表格外围使用深些的边框会更好一点）。

在样式化表格之前，先把表格放到Tony的网页上

在Tony的新表格中添加样式之前，我们得先把表格放到Tony的网页上。Tony的主页中有font-family和font-size，还有许多其他的样式等着表格继承。所以，如果不把表格放到Tony的网页中，我们就不知道表格在网页上显示的样子。

首先，在“chpter13/journal”文件夹中打开“journal.html”文件，以August 20th为入口，做以下的修改。完成修改后，转移到下一个等待加载的页面。

```
<h2>August 20, 2005</h2>
<p>
   <img src="images/segway2.jpg" alt="Me and my Segway in New Mexico" />
</p>

<p>
Well, I made it 1200 miles already, and I passed through some interesting
places on the way:
</p>

<ol>
   <li>Walla Walla, WA</li>
   <li>Magic City, ID</li>
   <li>Bountiful, UT</li>
   <li>Last Chance, CO</li>
   <li>Truth or Consequences, NM</li>
   <li>Why, AZ</li>
</ol>

<table summary="This table holds data about the cities I visited on my travels.  I've included
     the date I was in each city, the temperature when I was there, and altitude and population
     of each city.  I've also included a rating of the diners where I had lunch, on a
     scale from 1 to 5.">
   <caption>The cities I visited on my Segway'n USA travels</caption>
   <tr>
      <th>City</th>
      <th>Date</th>
      <th>Temperature</th>
      <th>Altitude</th>
      <th>Population</th>
      <th>Diner Rating</th>
   </tr>
   .
   .
   .
</table>
```

这是旧版的城市列表。删除后用一个表格代替。

这是新的表格。创建这个表格最简单的方法就是从上一个文件中复制和粘贴过来。

现在开始样式化表格

首先，我们需要将表格样式复制到"journal.css"文件中。我们要改变表格，那么就先用新的样式代替吧。下面，新的样式在样式文件的结尾部分突显出来。

```
body {
    font-family: Verdana, Geneva, Arial, sans-serif;
    font-size: small;
}
h1, h2 {
    font-weight: normal;
    color: #cc6600;
    border-bottom: thin dotted #888888;
}
h1 {
    font-size: 170%;
}
h2 {
    font-size: 130%;
}
blockquote {
    font-style: italic;
}


table {
    margin-left: 20px;
    margin-right: 20px;
    border: thin solid black;
    caption-side: bottom;
}

td, th {
    border: thin dotted gray;
    padding: 5px;
}

caption {
    font-style: italic;
    padding-top: 8px;
}
```

这是当前Tony网页的全部样式，都是在第9章中添加的。下面，我们将给表格添加新的样式。

首先，我们样式化表格。分别在表格的左、右两边添加边界及细的黑色边框。

接着将标题移到表格的下部。

改变数据单元格的边框，设置成较淡的点状的灰色边框。

然后在单元格中添加一些补白，使得单元格内容与边框之间留一些空隙。

这条规则格式化标题。将标题字体属性font-style改为斜体，并在顶部添加补白。

测试格式化后的表格

我们一次就做了这么多的改动，表格看上去立刻有了很大的变化。你应该校对一下改动的地方，确认并保存那些修改。然后，将journal.html载入你的浏览器。

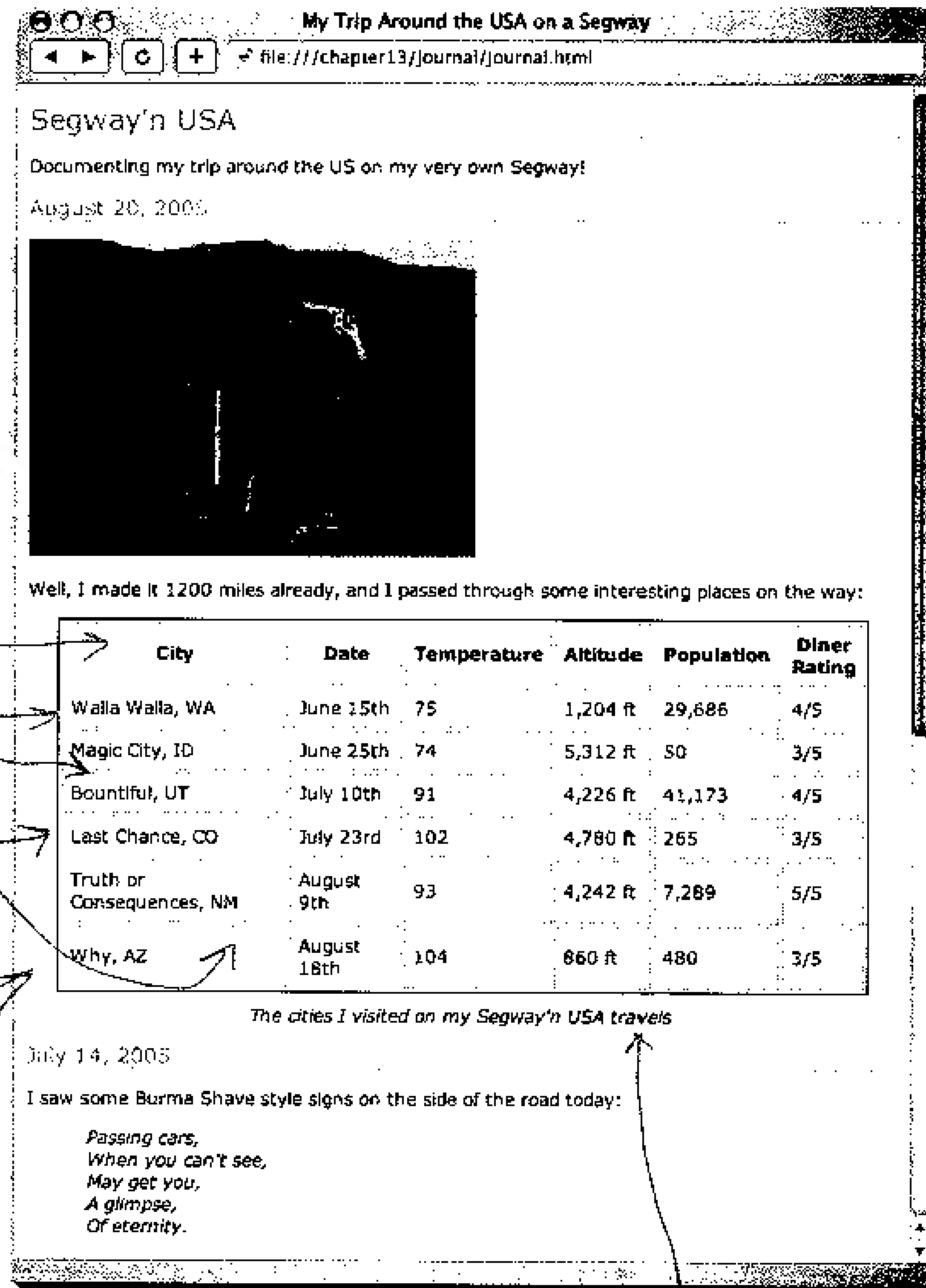
表格样式化后看上去大不相同了。我们还继承了一些已在Tony的日志中使用的样式。

所有的字体现在都是从前面已保存的文件样式中挑选出来的小号的sans-serif。

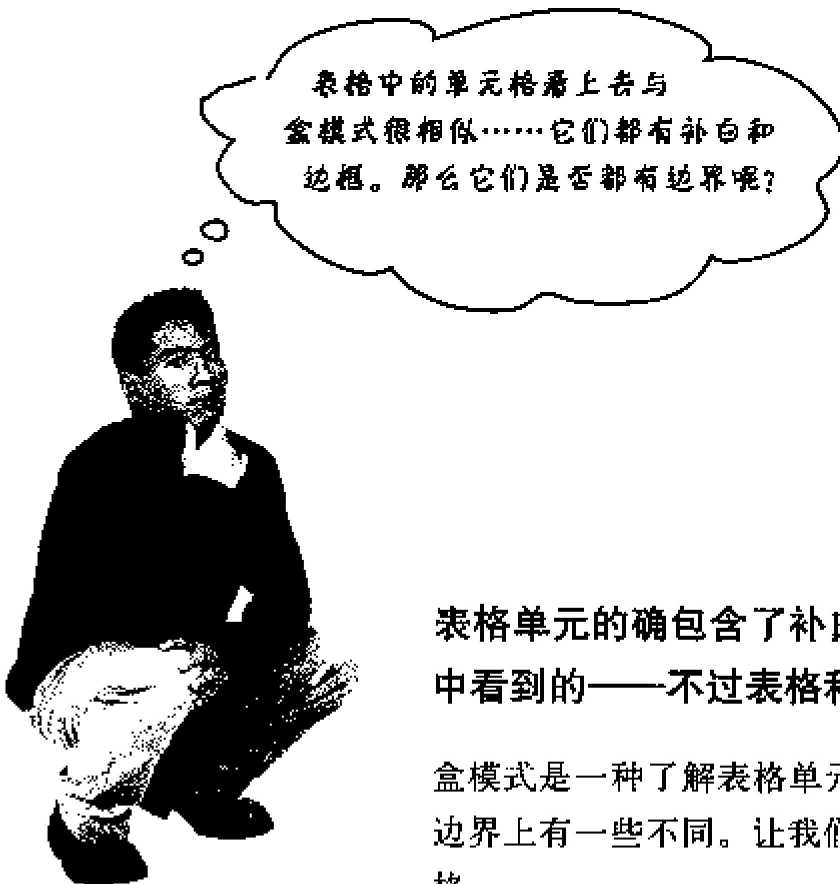
现在我们得到了一个深黑色的边框和虚线条。

我们在表格中添加了一些边界，在每个单元格中增添了一些补白。

虚线看上去复杂且分散，但对成对的单元格之间的复制不起任何作用。

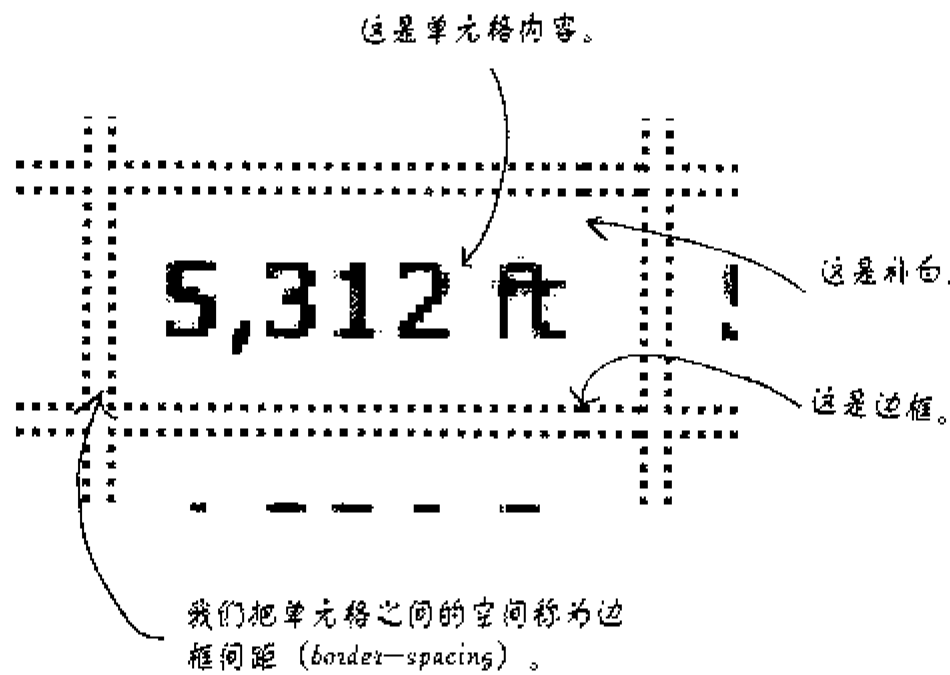


记住，在那些不支持caption-side属性的浏览器中，标题仍将位于表格的上方。



表格单元的确包含了补白和一个边框——正如你在盒模式中看到的——不过表格和盒模式在边界属性上有所不同。

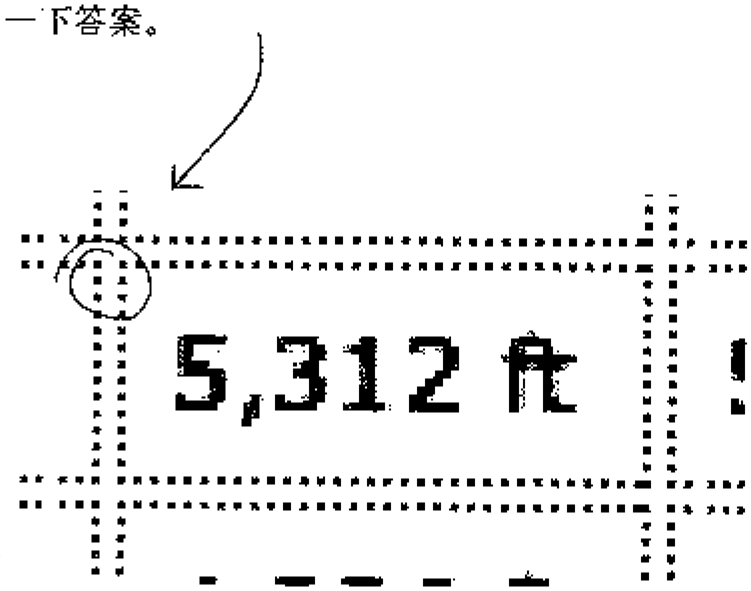
盒模式是一种了解表格单元的好方式，不过盒模式与表格单元在边界上有一些不同。让我们先来看一看，Tony表格中的一个单元格：



我们使用边框间距属性来代替边界，边框间距属性定义在整个表格中。换句话说，我们不能对单个的表格单元而只能对所有的单元格设置边界属性。

Sharpen your pencil

双虚线让Tony的表格看上去复杂而又分散。如果每个表格单元只有一个边框的话，表格看上去会更好而且并不分散。你能够运用所学的知识想出一个办法将双虚线变为单虚线吗？试一试，接着在本章后核对一下答案。



there are no Dumb Questions

问：所以边框间距是相对于整个表格的，而边界只是针对于单个的元素？

答： 没错。表格单元不存在边界，它只有边框四周的空间而已，而这个空间是为整个表格设置的。你不能单独控制单个表格单元的边框间距。

问：有没有什么办法设置不同的水平和垂直方向上的边框间距？那看上去很有用。

答： 当然有了。你可以在你的边框间距中这样定义：

```
border-spacing:10px30px;
```

那条语句设置了10个像素的水平边框间距和30个像素的垂直边框间距。

问：边框间距属性在我的浏览器中好像不起作用啊。

答： 你使用的是Internet Explorer浏览器吗？很遗憾地告诉你，IE 6不支持边框间距属性。抱歉我们没有提早告诉你。不过，以后你不会忘记你的浏览器不支持边框间距属性，对吧？

压缩边框

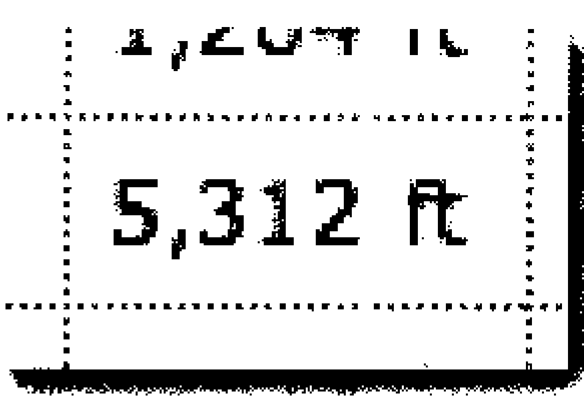
除**border-spacing**属性之外，还有另外一种方法可以处理边框空间问题。你可以使用CSS中一个叫做**border-collapse**的属性来消除边框间距。使用这个属性时，浏览器将忽略设置在表格上的所有边框间距，并且将两个相邻的边框合并为一个边框。这样就将两个边框“压缩”为一个边框了。

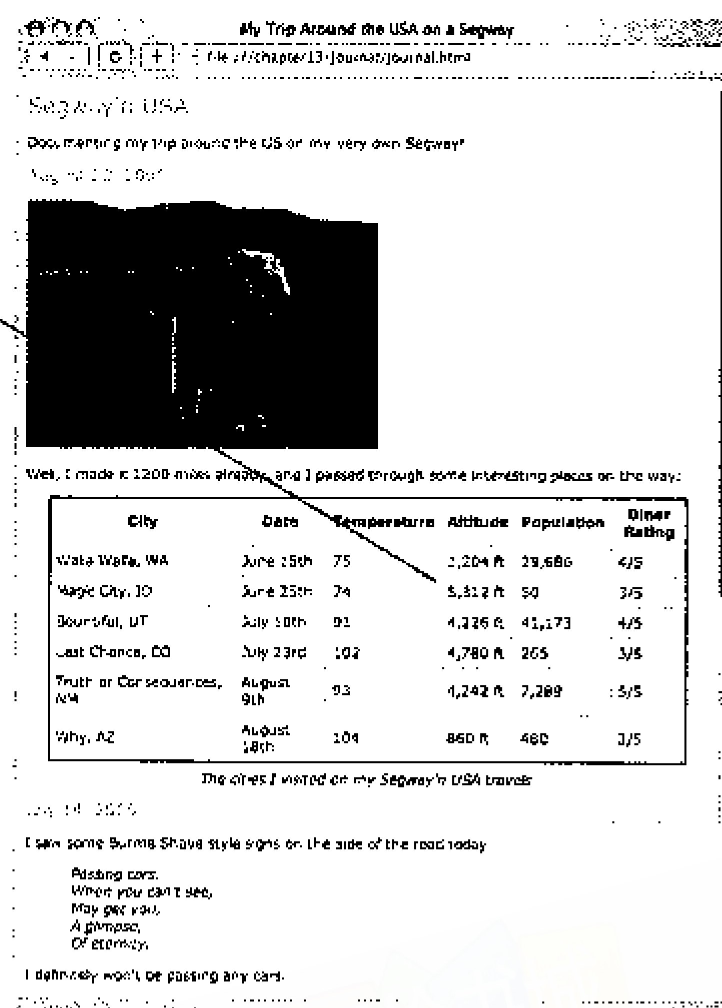
像下面这样设置**border-collapse**属性。照着下面的例子，在你的“journal.css”文件中做如下的修改：

```
table {
    margin-left: 20px;
    margin-right: 20px;
    border: thin solid black;
    caption-side: bottom;
    border-collapse: collapse;
}
```

添加一个border-collapse属性，设置其值为“collapse”。

保存这个文件并加载页面，然后检查边框的变化。





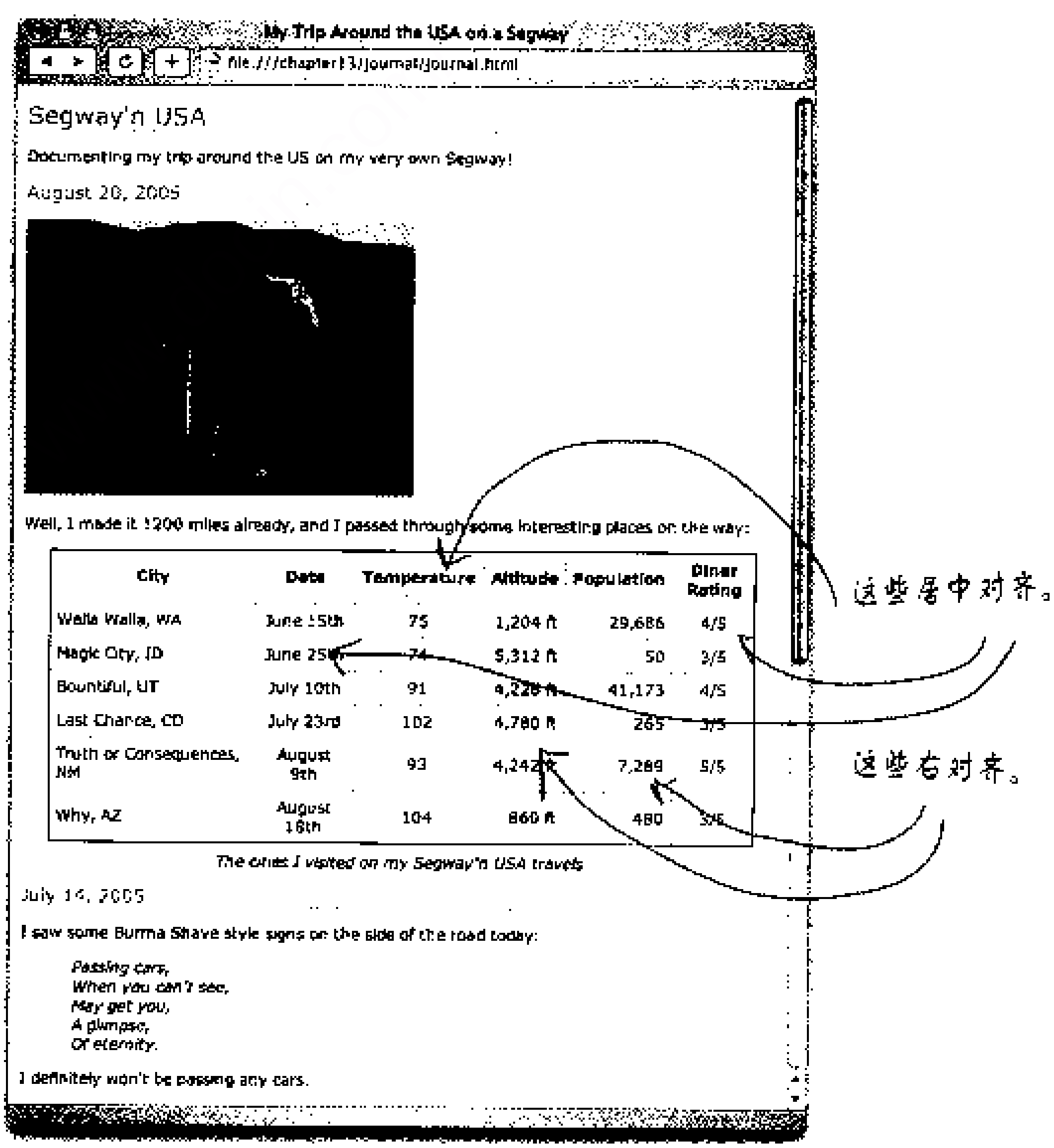
现在，你所看到的是所有的表格单元都是由单边框环绕。这正是我们所想要的效果，你不觉得表格现在看上去更加清晰了吗？

Sharpen your pencil

在XHTML和CSS方面你已经很专业了，所以，我们不介意让你在下面的练习中稍微练练手。这个如何：以文本对齐方式为出发点，再次修饰表格。我们想将日期、温度、就餐率在单元格中居中对齐，海拔高度和人口右对齐。那么，你该怎样去做呢？

提示：创建两个类，一个类为center-aligned,一个类为right-aligned。在每个类中使用text-align属性。最后，添加到适当的类中修正<td>元素。

这道题听起来很困难，不过只有一步一步解决了。你已经掌握了完成此题的相关知识。还有，那就是你可以在这一章后面找到正确答案，不过看之前你还是花点时间自己做一做。

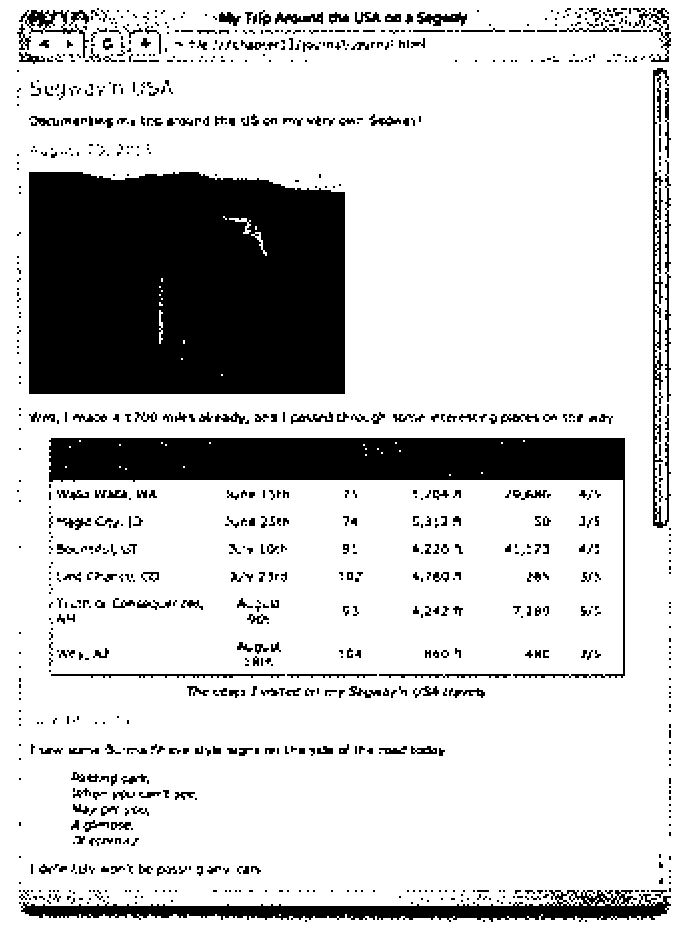


对于颜色呢？

你知道Tony喜欢彩色的文字。那么没有理由不在他的表格中添加一些颜色。这样不仅看起来更加美观，而且还能提高表格的可读性。就像其他元素一样，你所要做的就是为每一个表格单元设置**backgound-color**属性来改变它的颜色(注意将你所学到的XHTML和CSS知识融会贯通)。下面就是你要做的：

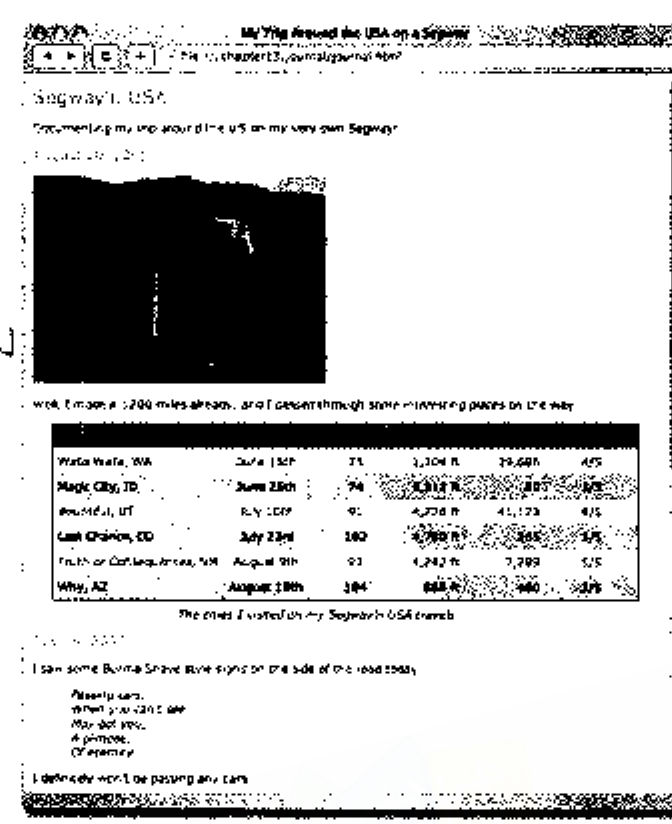
```
th {
    background-color: #cc6600;
}
```

把这条新的规则添加到你的“journal.css”文件中并加载网页。显示效果如图所示：



给表格行添加颜色看上去如何？

添加颜色后表格看上去更漂亮了。那么我们开始下一步。给表格添加颜色最常用的方式就是行行交替添加颜色，这样你就可以很容易看清楚每一行而不至于混淆了某列的某行。验证一下：



在CSS中添加颜色很难吗？不是的。下面让我们来看一看添加颜色是如何实现的。首先，定义一个新类，命名为“cellcolr”：

```
.cellcolor {
    background-color: #fcba7a;
}
```

然后在你想要添加颜色的表格行中添加这个类的属性。那么，你将会看到Magic City，Last Chance，Why等行的**<tr>**开始标签中添加了**class="cellcolor"**属性。

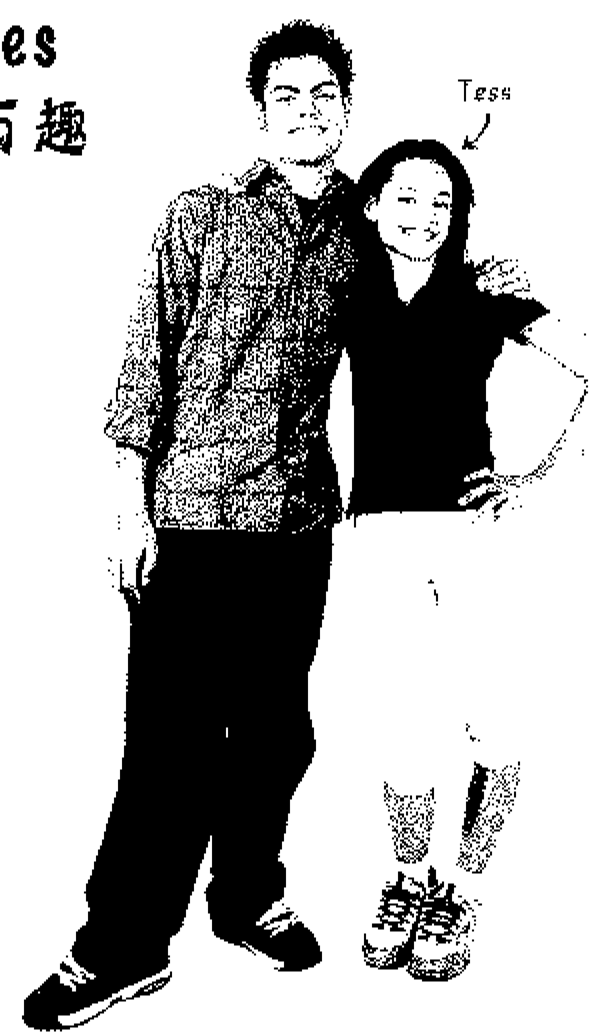
练习

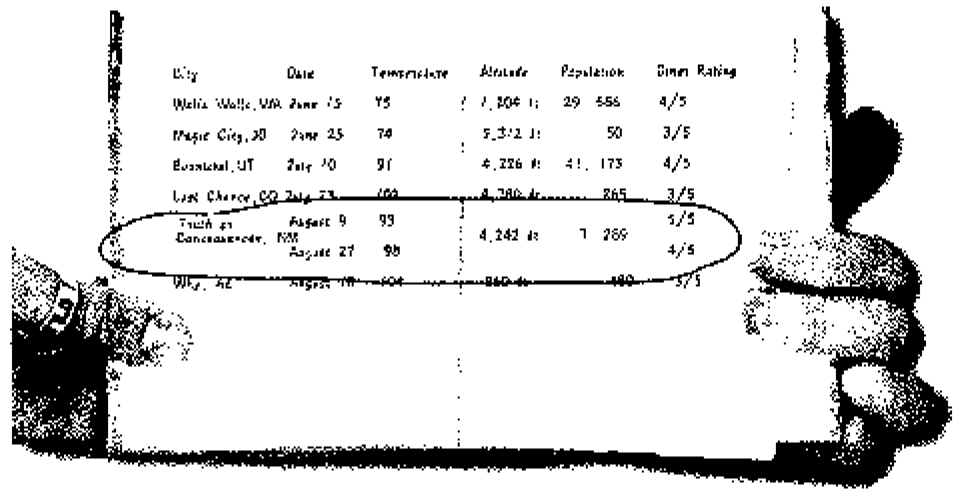
轮到你做练习了。使用CSS把“cellcolor”类添加到“journal.css”文件中，然后用XHTML编写，在需要交替添加颜色的行的<tr>开始标记中，添加class=“cellcolor”规则。继续下面的学习之前，核对一下你的答案。

不知我们有没有注意到，Tony在Truth or Consequences NewMexico一行中有一个有趣的发现。

准确地说应该是Tony在Truth or Consequences New Mexico镇，发现了有趣的东西。实际上是，Tony发现Tess去过Arizona城后，还想去。于是他们逛了一大圈之后又回到了Mexico镇。

我们是不是应该感谢Tony,他给我们提出了表格上的一个难题。我们可以另起一行来添加Truth or Consequences的信息，不过我们更愿意用“文雅”一点儿的方式添加信息。什么是“文雅”一点儿的方式？翻开下一页，你就能找到答案。





从另一个角度看Tony的表格

基于Tony再次回到New Mexico镇，就在原先Truth or Consequence的下边，Tony添加了一个叫做August 27th的新人口。他重复使用了两个相同的单元格（一种在单元格中减少信息的好技巧），你可以看到当他添加新的一行时，他需要做的只是列出和第二次旅行时不同的数据（日期，温度，再次访问时的就餐率）。

这是Tony在Truth or Consequences镇的两次旅行。

<table>
<tr><th>City</th><th>Date</th><th>Temp</th><th>Altitude</th><th>Population</th><th>Diner Rating</th></tr>
<tr><td>Walla Walla, WA</td><td>June 15th</td><td>75°</td><td>1,204 ft</td><td>29,686</td><td>4/5</td></tr>
<tr><td>Magic City, ID</td><td>June 25th</td><td>74°</td><td>5,312 ft</td><td>50</td><td>3/5</td></tr>
<tr><td>Bountiful, UT</td><td>July 10th</td><td>91°</td><td>4,226 ft</td><td>41,173</td><td>4/5</td></tr>
<tr><td>Last Chance, CO</td><td>July 23rd</td><td>102°</td><td>4,780 ft</td><td>265</td><td>3/5</td></tr>
<tr><td rowspan="2">Truth or Consequences, NM</td><td>August 9th</td><td>93°</td><td rowspan="2">4,242 ft</td><td rowspan="2">7,289</td><td>5/5</td></tr>
<tr><td>August 27th</td><td>98°</td><td>4/5</td></tr>
<tr><td>Why, AZ</td><td>August 18th</td><td>104°</td><td>860 ft</td><td>480</td><td>3/5</td></tr>
</table>

这些表格行跨越了两行。

不过在何处用XHTML处理上面留下的问题呢？看起来，你不得不另外添加一行了，仅仅只需要复制城市、海拔、人口等表格数据，对吗？不要着急。我们另有办法……使用XHTML表格，你可以将单元格扩展为多行（或者多列），让我们看看是如何实现的。

将单元格扩展为多行

将一个单元格扩展为多行是什么意思呢？让我们先看看Tony表格中的Truth or Consequences, NM入口。数据单元格中的城市、海拔、人口扩展为两行而不是一行，相对的日期、温度、就餐率扩展为一行，这是数据单元格普遍默认的格式。

City	Date	Temp	Altitude	Population	Diner Rating
Walla Walla, WA	June 15th	75°	1,204 ft	29,686	4/5
Magic City, ID	June 25th	74°	5,312 ft	50	3/5
Bountiful, UT	July 10th	91°	4,226 ft	41,173	4/5
Last Chance, CO	July 23rd	102°	4,780 ft	265	3/5
Truth or Consequences, NM	August 9th	93°	4,242 ft	7,289	5/5
	August 27th	98°			4/5
Why, AZ	August 18th	104°	860 ft	480	3/5

这些单元格扩展为两行。

而日期、温度、就餐率等单元格占一行。

那么，如何用XHTML实现多行跨越呢？它的实现比你想象中还要简单。你可以使用**rowspan**属性定义一个表格单元所占据的行数。然后将相同的表格数据元素从其他扩展行中消除。看看吧——看例子比文字描述更容易理解：

```
<tr>
   <td rowspan="2">Truth or Consequences, NM</td>
   <td class="center">August 9th</td>
   <td class="center">93</td>
   <td rowspan="2" class="right">4,242 ft</td>
   <td rowspan="2" class="right">7,289</td>
   <td class="center">5/5</td>
</tr>
<tr>
                                  }

   <td class="center">August 27th</td>
   <td class="center">98</td>
                                  }
                                  }
   <td class="center">4/5</td>
</tr>
```

这里，New Mexico单元格占据了两行的空间。

在第二次旅行时没有改变的那些数据单元格（城市，海拔，人口）我们添加了一个rowspan属性，用来将表格单元扩展为两行。

由于只是扩展，所以没有必要写城市。

在第二行中，我们只定义了那些需要扩展的单元格（日期，温度，新的就餐率）

海拔和人口也一样没必要写。

连连看

确认一下你已经掌握了这些知识。那么，在每个<td>元素到对应的表格单元之间画一条带箭头的线。继续学习之前，核对一下答案。

```
<tr>
   <td rowspan=“2”>Truth or Consequences, NM</td>
   <td class=“center”>August 9th</td>
   <td class=“center”>93</td>
   <td rowspan=“2” class=“right”>4,242 ft</td>
   <td rowspan=“2” class=“right”>7,289</td>
   <td class=“center”>5/5</td>
</tr>
<tr>

   <td class=“center”>August 27th</td>
   <td class=“center”>98</td>

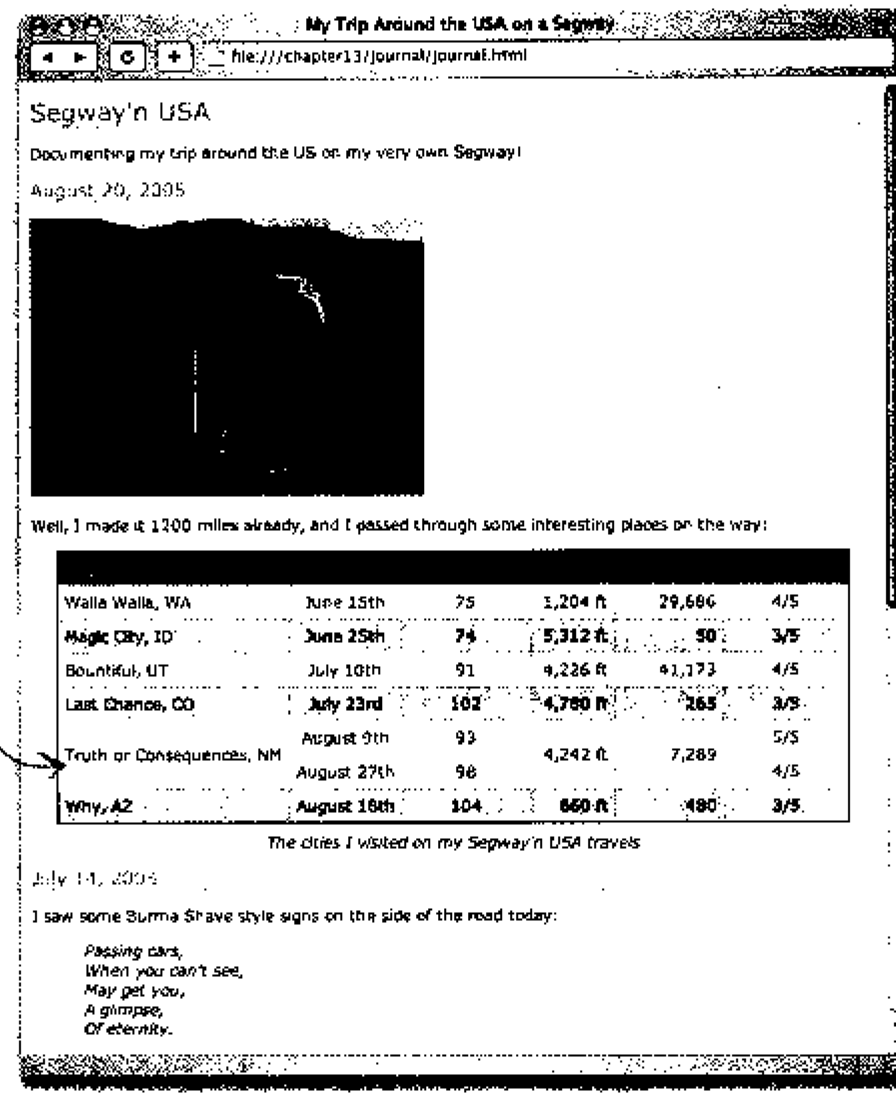

   <td class=“center”>4/5</td>
</tr>
```

Truth or Consequences, NM	August 9th	93°	4,242 ft	7,289	5/5
	August 27th	98°			4/5

新型的改进了的表格

将改动的地方保存到“journal.html”文件中，测试一下结果。再看看表格的显示效果。仔细想一想，你是如何修改表格的：使用XHTML定义一些需要占据多行的单元格，要实现定义，需要将**<td>**标记从原来的位置消除。

现在我们得到了一个看起来很不错的，没有冗余信息的表格。



www.docin.com

there are no Dumb Questions

问：你说也可以把表格数据单元格扩展为多列，是吗？

答： 是的，当然可以。在<td>元素中添加colspan属性并设置列的数值。不同于rowspan属性，colspan属性在扩展多列时，你需要消除表格单元中那些相同的行元素（因为扩展的是多列而不是多行）。

问：我可以在同一个<td>元素中添加colspan和rowspan两个属性吗？

答： 是的，当然可以。只需保证调整表格中的其他<td>元素来说明扩展的行和列就可以了。换句话说就是，你需要从同一行和同一列中消除那些相同的<td>标签。

问：你认为这些行扩展看上去真得很美观吗？

答： 当然。行扩展减少了表格中一定数量的信息，通常来说这是一件好事。还有，在现实世界中随便看看，你会发现行扩展和列扩展非常普遍，所以用XHTML实现行扩展和列扩展很有用。不过，如果你喜欢原来的表格，你可以随意改变你的XHTML文件，回到以前版本的表格。

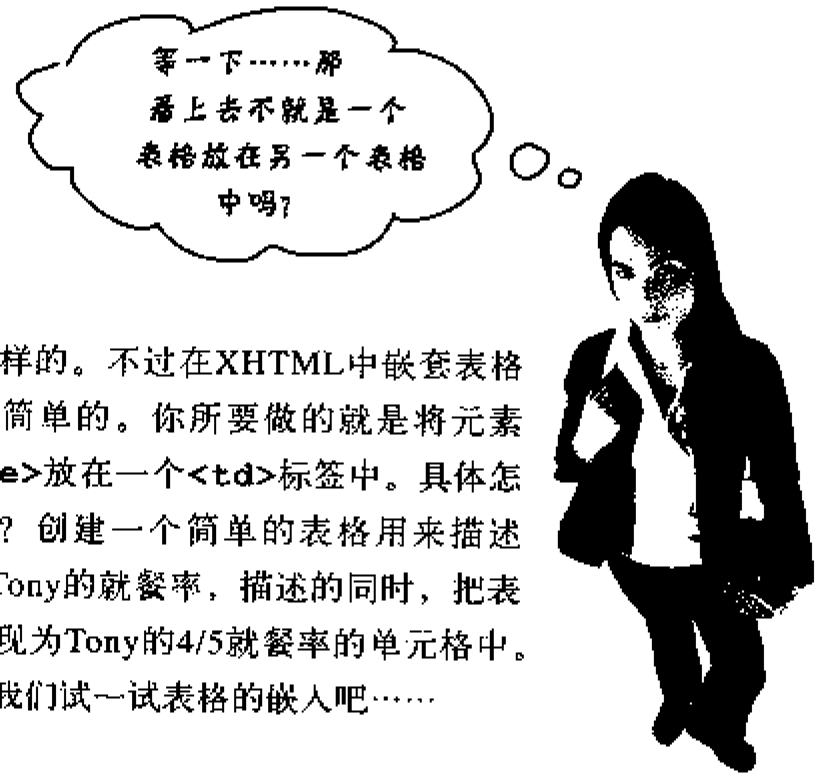


天堂的烦恼？

看上去，在Auguest 27th的就餐率上，要么保持Tony和Tess观点不同，要么说服他们相同，我们该怎么办呢？我们需要在表格中增添一个就餐率吗？我们不想只因为Tess的观点而另外增添一个入口。那么……我们为什么不这么办呢？

<table>
<tr><th>City</th><th>Date</th><th>Temp</th><th>Altitude</th><th>Population</th><th>Diner Rating</th></tr>
<tr><td>Walla Walla, WA</td><td>June 15th</td><td>75°</td><td>1,204 ft</td><td>29,686</td><td>4/5</td></tr>
<tr><td>Magic City, ID</td><td>June 25th</td><td>74°</td><td>5,312 ft</td><td>50</td><td>3/5</td></tr>
<tr><td>Bountiful, UT</td><td>July 10th</td><td>91°</td><td>4,226 ft</td><td>41,173</td><td>4/5</td></tr>
<tr><td>Last Chance, CO</td><td>July 23rd</td><td>102°</td><td>4,780 ft</td><td>265</td><td>3/5</td></tr>
<tr><td rowspan="2">Truth or Consequences, NM</td><td>August 9th</td><td>93°</td><td rowspan="2">4,242 ft</td><td rowspan="2">7,289</td><td>5/5</td></tr>
<tr><td>August 27th</td><td>98°</td><td><table><tr><td>Tess</td><td>5/5</td></tr><tr><td>Tony</td><td>4/5</td></tr></table></td></tr>
<tr><td>Why, AZ</td><td>August 18th</td><td>104°</td><td>860 ft</td><td>480</td><td>3/5</td></tr>
</table>

我们为什么不把他们的就餐率放在一个表格中呢，那样一来，我们就可以得到更加精确的信息了。



就是那样的。不过在XHTML中嵌套表格是非常简单的。你所要做的就是将元素<table>放在一个<td>标签中。具体怎么做呢？创建一个简单的表格用来描述Tess和Tony的就餐率，描述的同时，把表格放在现为Tony的4/5就餐率的单元格中。那么，我们试一试表格的嵌入吧……

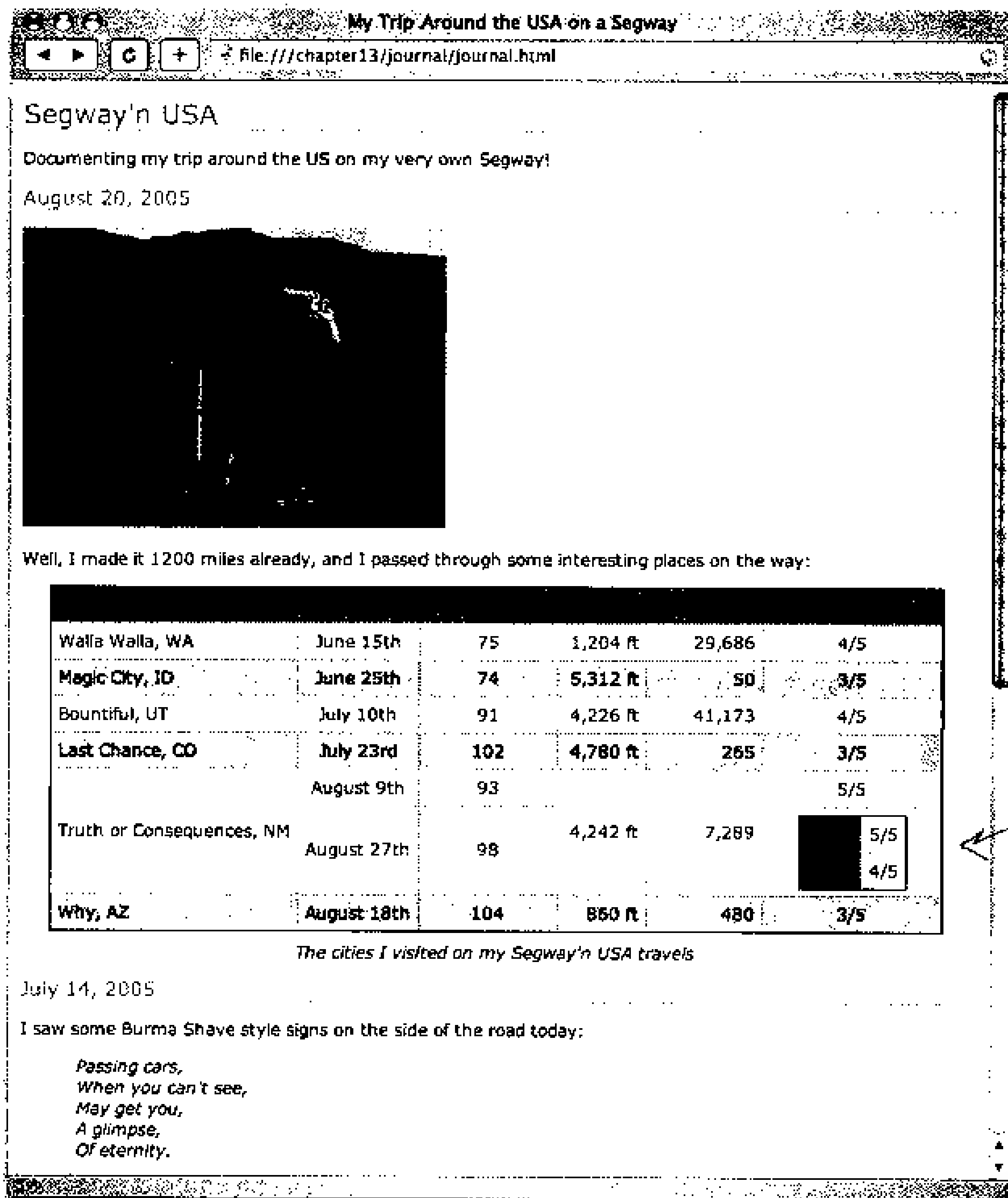
```
<tr>
   <td rowspan="2">Truth or Consequences, NM</td>
   <td class="center">August 9th</td>
   <td class="center">93</td>
   <td rowspan="2" class="right">4,242 ft</td>
   <td rowspan="2" class="right">7,289</td>
   <td class="center">5/5</td>
</tr>
<tr>
   <td class="center">August 27th</td>
   <td class="center">98</td>
   <td>
     4/5
     <table>
        <tr>
           <th>Tess</th>
           <td>5/5</td>
        </tr>
        <tr>
           <th>Tony</th>
           <td>4/5</td>
        </tr>
     </table>
   </td>
</tr>
```

首先，删除原来Tony的就餐率……

……然后，在此处用一个表格代替。这个表格包含了两个就餐率，一个是Tess的，一个是Tony的。我们用他们的名字来命名表格的表头，数据单元为就餐率。

测试运行嵌套表格

那么，我们开始将新的表格输入到文件中。表格的输入很容易出错，在确认无误后加载到网页上。你将看到一个新的嵌套的表格。



噢，看上去真漂亮。嗯，嵌套表格的背景颜色看上去不是很好。让我们保留粗体的名字，去掉背景颜色。

BRAIN BARBELL

回顾一下你所练习的题目。你所要做的就是改变Tony和Tess单元格的表格表头的背景颜色，同时不能改变主表格表头的背景颜色。怎样实现它？只要使用一个只选中嵌套表格表头的选择符就可以了。

Walla Walla, WA	June 15th	75	1,204 ft	29,686	4/5
Magic City, ID	June 25th	74	5,312 ft	50	3/5
Bountiful, UT	July 10th	91	4,226 ft	41,173	4/5
Last Chance, CO	July 23rd	102	4,780 ft	265	3/5
Truth or Consequences, NM	August 9th	93	4,242 ft	7,289	5/5
	August 27th	98			Tess 5/5 Tony 4/5
Why, AZ	August 18th	104	860 ft	480	3/5

The cities I visited on my Segway'n USA travels

我们希望把嵌套的表格表头的背景颜色换成白色。

确定只选中表格表头元素的选择符。

```
________________ {
    background-color: white;
}
```



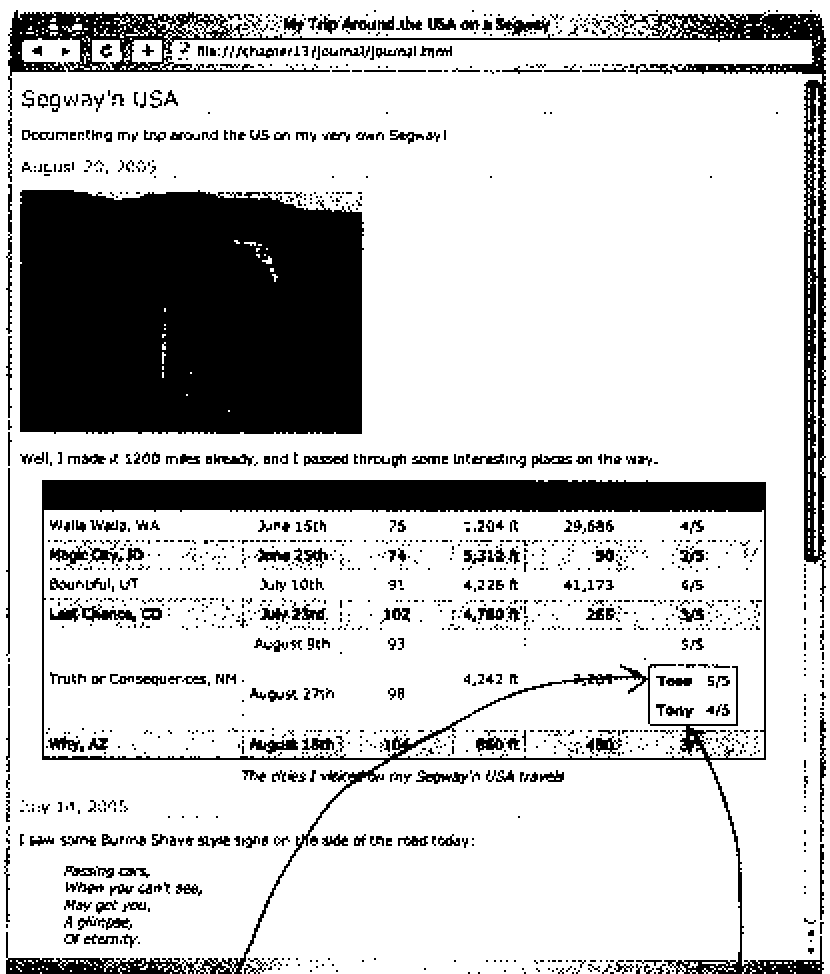
停下来！直到你完成了这个练习，你可以翻到下一页。

为嵌套的表格表头覆盖CSS

以<th>元素为对象,在嵌套的表格中使用子孙选择符。在你的CSS程序中使用“table table th”选择符添加一条新的规则来改变嵌套的表格表头的背景颜色为白色:

```
table table th {
    background-color: white;
}
```

现在保存这些改动到你的“journal.css” 文件中并重新加载。



现在，嵌套的表格表头背景换成了白色。

但是，注意到这个单元格仍然使用粗体字体，那是因为我们并没有覆盖字体属性。

there are no Dumb Questions

问： 我使用一个类来解决“灵机一动”中的问题。我还创建了一个名为“nestedtable”的类，并为其指定每一个表头，接着创建了像这样的一条规则：

```
.nestedtable {
            background-color:
white;
}
```

那样的规则能解决该问题吗?

答： CSS中有很多种解决此问题的方法。你的方法也不乏为一个有效的、完美的、正确的途径。我们不可能将解决的方法一一列出来，这里我们仅仅列出了使用选择符的方法。如果Tony和Tess仍然坚持添加各自的就餐率的话，我们该怎么办呢？那么，我们只需要针对每一个就餐率在每个<th>元素中添加此类。在这种方式下，样式化将自动进行。

BRAIN POWER

你希望Tony和Tess在他们各自的表格行中设置不同的背景颜色，比如说蓝色和粉红色。你能够想出几种方法呢?

Tony网站的最后润饰

Tony的网页看起来真的很漂亮了，不过网页上还有一处地方我们没有花时间进行样式化：Tony为他的旅行所准备的包含一系列选项的列表。在Tony的June 2nd入口，你会找到这个列表。下面我们就来看一看：

看看，来自June 2nd入口的一个XHTML选段。

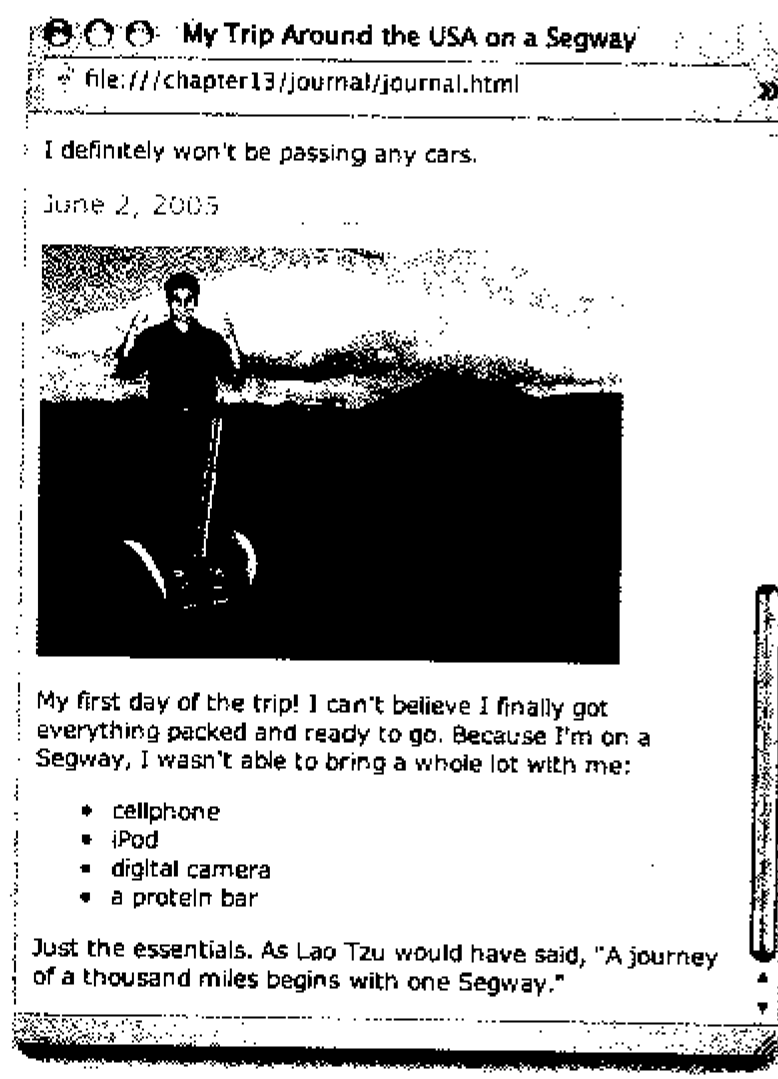
```
.
.
.
  <h2>June 2, 2005</h2>

  <p>
    <img src="images/segway1.jpg"
         alt="The first day of the trip" />
  </p>

  <p>
    My first day of the trip! I can't
    believe I finally got everything
    packed and ready to go. Because
    I'm on a Segway, I wasn't able
    to bring a whole lot with me:
  </p>
  <ul>
    <li>cellphone</li>
    <li>iPod</li>
    <li>digital camera</li>
    <li>a protein bar</li>
  </ul>
  <p>
    Just the essentials.  As Lao Tzu
    would have said, <q>A journey of
    a thousand miles begins with
    one Segway.</q>
  </p>
</body>
</html>
```

这是Tony日志的底部，也就是Jounal.html的末尾部分。想起Tony第一篇日志入口的那个列表了吗？

显示出来的列表就是这样的。



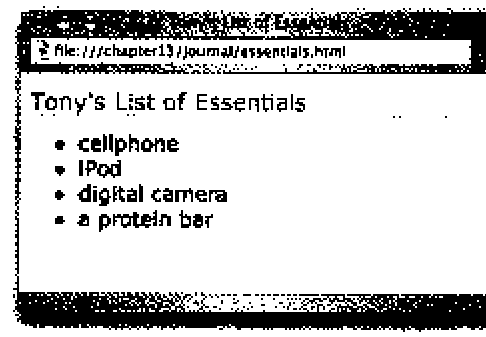
一些列表格式

你已经掌握了你所知道的基础的CSS字体、文本、颜色及其他的属性，那么你可以样式化所有内容了。当然，你同样可以格式化列表。实际上，只有几个用来定义列表的属性。列表的主属性为“**list-style-type**”，此属性允许在你的列表中控制bullet（或者和他们一样，称为标志“marker”）。下面介绍几种列表样式：

这里，我们在<li>元素中设置了样式。你也可以在<ul>元素中设置，元素<li>将继承<ul>属性。

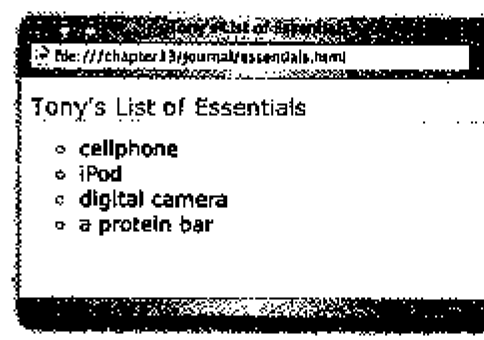
```
li {
    list-style-type: disc;
}
```

默认的情况下，标志的类型为圆点。



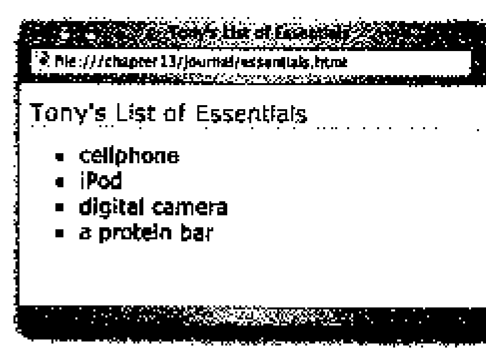

```
li {
    list-style-type: circle;
}
```

圆圈属性值设置一个简单的圆圈形状标志。



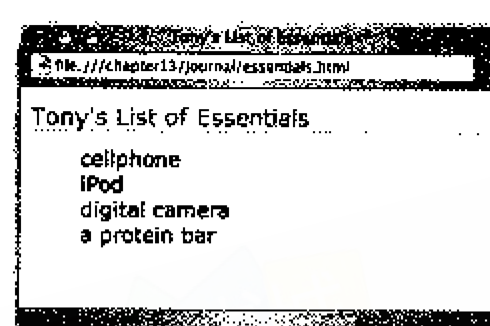

```
li {
    list-style-type: square;
}
```

方块属性值设置方块标志。




```
li {
    list-style-type: none;
}
```

None值消除所有标志。



如果你想要一个自定义的标志，该怎么办？



你真的更希望使用自己喜欢的标志吗？噢，运气很好。在CSS中一个叫做**list-style-image**的属性允许你把图像作为列表的标志。让我们在的Tony列表上试一下：

这就是设置到URL上的list-style-image属性。

```
li {
    list-style-image: url(images/backpack.gif);
    padding-top: 5px;
    margin-left: 20px;
}
```

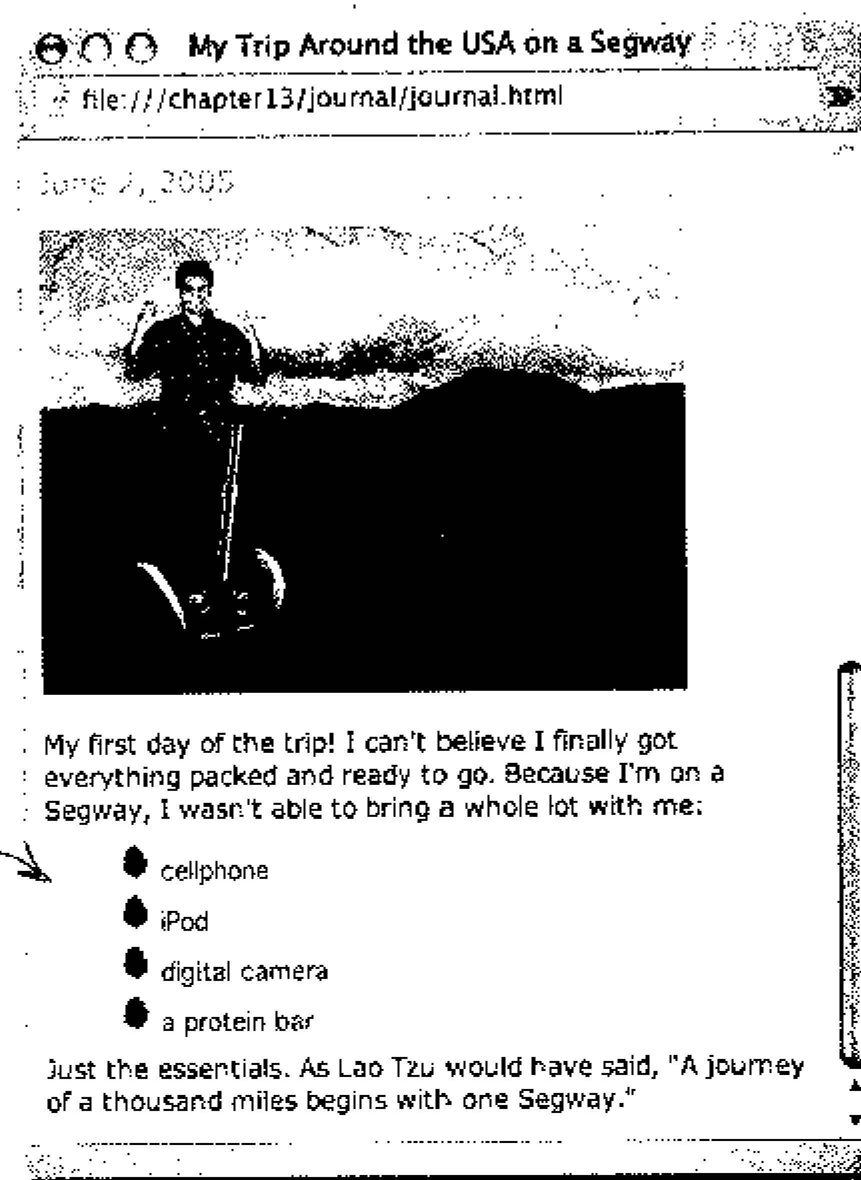
"backpack.gif"图像是书包的缩小版。看起来很合适，对吧；而且还是Tony署名的颜色哦。

我们在列表选项的左边添加了一些边界属性，用来增加一些空间，同时增加一些顶部补白属性，用来设置列表选项的头部空间。

最终测试……

这就是你对Tony网站的最后改动。把列表项的规则添加到你的CSS中，并重新加载到网页上。

此处，用图标替代了原来的标志，而且增加了额外的边距和间距。



there are no Dumb Questions

问： 什么是顺序列表？怎样改变它的样式？

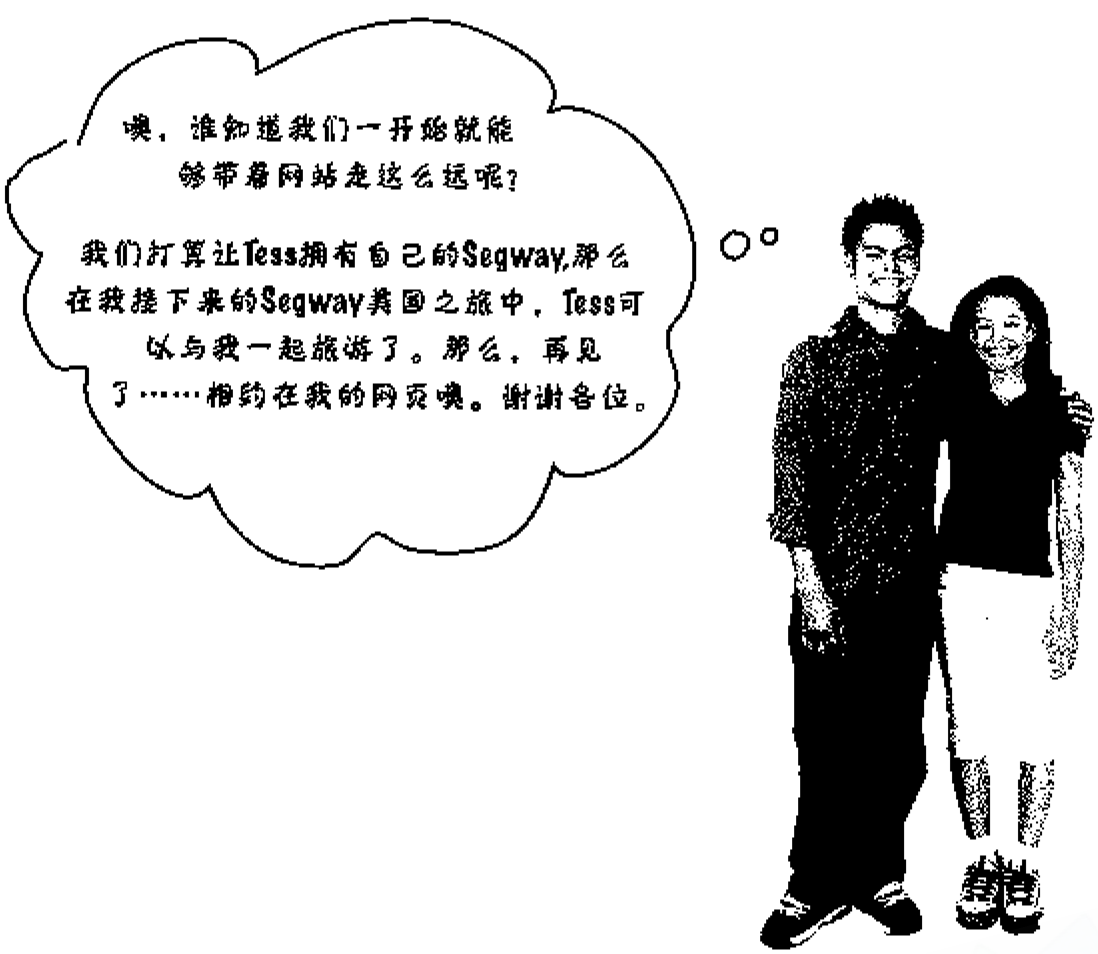
答： 你可以用同样的办法样式化顺序列表和非顺序列表。顺序列表有一组标志，这一组标志由数字或字母排列而成，用CCS的list-style-type属性你可以控制顺序列表，不管它是十进制数字、罗马数字还是字母表字母（例如，a,b,c）。通常list-style-type属性值的取值为十进制数、大写字母、小写字母、大写罗马数字或小写罗马数字。参阅一下CSS使用指南，上面有很多种取值。

问： 如何控制列表的文本环绕？也就是说，如何控制让文本环绕在标记的下面？还是标记环绕在文本的下面？

答： 有一个叫做list-style-position的属性可以处理这个问题。将此属性值设置为inside,那么文本就环绕在标记的下面；设置为outside，那么标记就环绕在文本的下面。

问： 你确定那是正确的吗？看上去还有点疑问。

答： 没有错，是正确的。inside和outside真正的含义是：当你把list-style-position设置为inside，那么标记位于列表选项中，所以文本环绕在标记的下面。当你把list-style-position设置为outside时，那么标记位于列表选项的外面，所以标记环绕在文本下面。inside your item的意思就是说在列表选项盒边框里面。



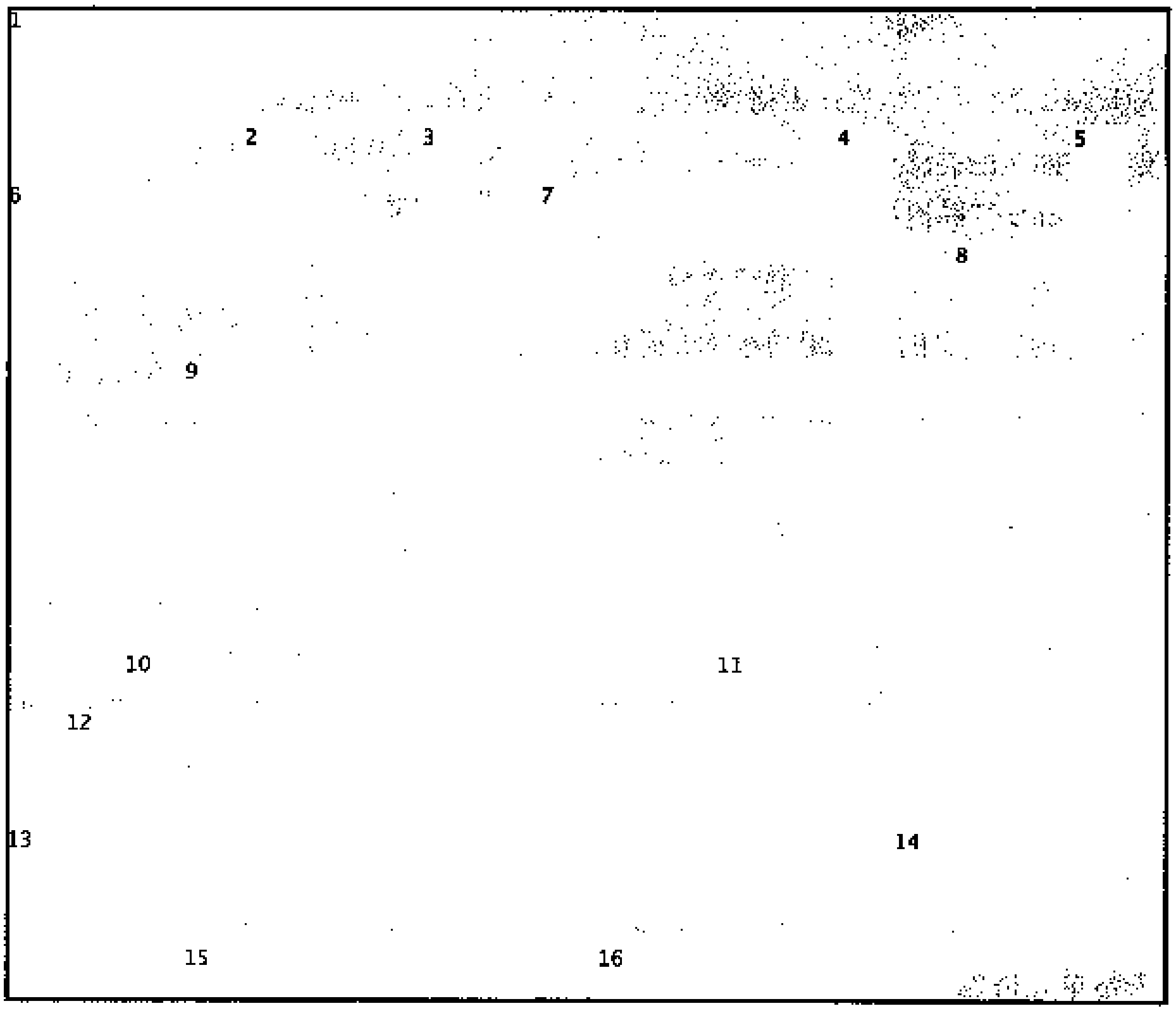
要点

- XHTML表格可以结构化表格式数据。
- 使用HTML的表格元素，<table>，<tr>，<th>，<td>等标记创建表格。
- <table>标记的定义作用于整个表格。
- 表格是由行定义的，使用的是<tr>元素。
- 每行中，使用<td>元素定义一个或多个单元格。
- 使用<th>标记定义一个数据单元格的行表头或者列表头。
- 表格以格子的方式罗列出来，每一行对应HTML中的一个<tr></tr>标记对，每一列对应行中<td></td>标记对之间的内容。
- 使用表格summary属性和<capition>元素，你可以提供一些表格的附加信息。
- 表格单元格包含补白、边框及单元格之间的边框间距。
- 正如你能控制元素的补白、边框、边界等属性一样，你也可以用CSS控制表格的补白、边框、边框间距等属性。
- Border-collapse属性是一个针对表格的特殊的CSS属性，它允许你将多个单元格合并为一个更清晰的单元格。
- 使用CSS中的text-align和verticalalign属性可以改变表格数据的对齐方式。
- 你可以使用background-color属性为你的表格添加一些颜色。背景颜色的设置可以是整个表格、一行或者单个单元格。
- 如果一个数据单元格为空，那么<td>元素中不放置任何内容。不管怎样，你都需要使用一个<td></td>标记对维持表格的对齐方式。
- 如果你的单元格需要被扩展为多行或者多列，你可以使用<td>元素的rowspan或者colspan属性。
- 在表格中嵌入一个表格，只需将表格及其所有的内容放进被嵌套表格的一个单元格中就可以了。
- 表格可以用来记录表格式的数据，而不是用来做网页的版面设计。如第12章中描述的那样，使用CSS定位创建多栏网页布局。
- 列表可以像其他元素那样被CSS样式化。CSS中有很多针对列表的属性，例如，list-style-type和 list-tyle-image。
- list-style-type改变列表中标志（marker）的类型。
- list-tyle-image定义图像列表标记。



填字游戏

交叉单词看上去有一点像表格，不是吗？开动你的脑筋，完成这个游戏所有的单词 都可从本章中找到。



横排提示：

1. 为屏幕读取器提供的表述。

6. 一个表格放入另一个表格叫做______。

9. 使用这个属性（用图像代替列表中构建的标志）。

12. 用来合并边框。

13 能够控制标志，是放在列表项边框里面还是外面。

15 我们把bullets叫做一个列表类型______。

16. 这儿需要一个<td>标记。

竖排提示：

1. 一个数据单元格用作多行或者多列时，它的操作名称。

2. 这儿需要一个<th>标记。

3. 能够改变列表标志的属性。

4. 表格包括补白、边框，但不包括_______。

5. 边框之间的区域名称。

7. 用来显示表格所添加的简短的表述。

8. list-item-position能控制文本的_______。

10. 用_____定义XHTML表格，不是用列。

11. 这种情况不能使用表格。

14. 标题默认的位置。

练习答案

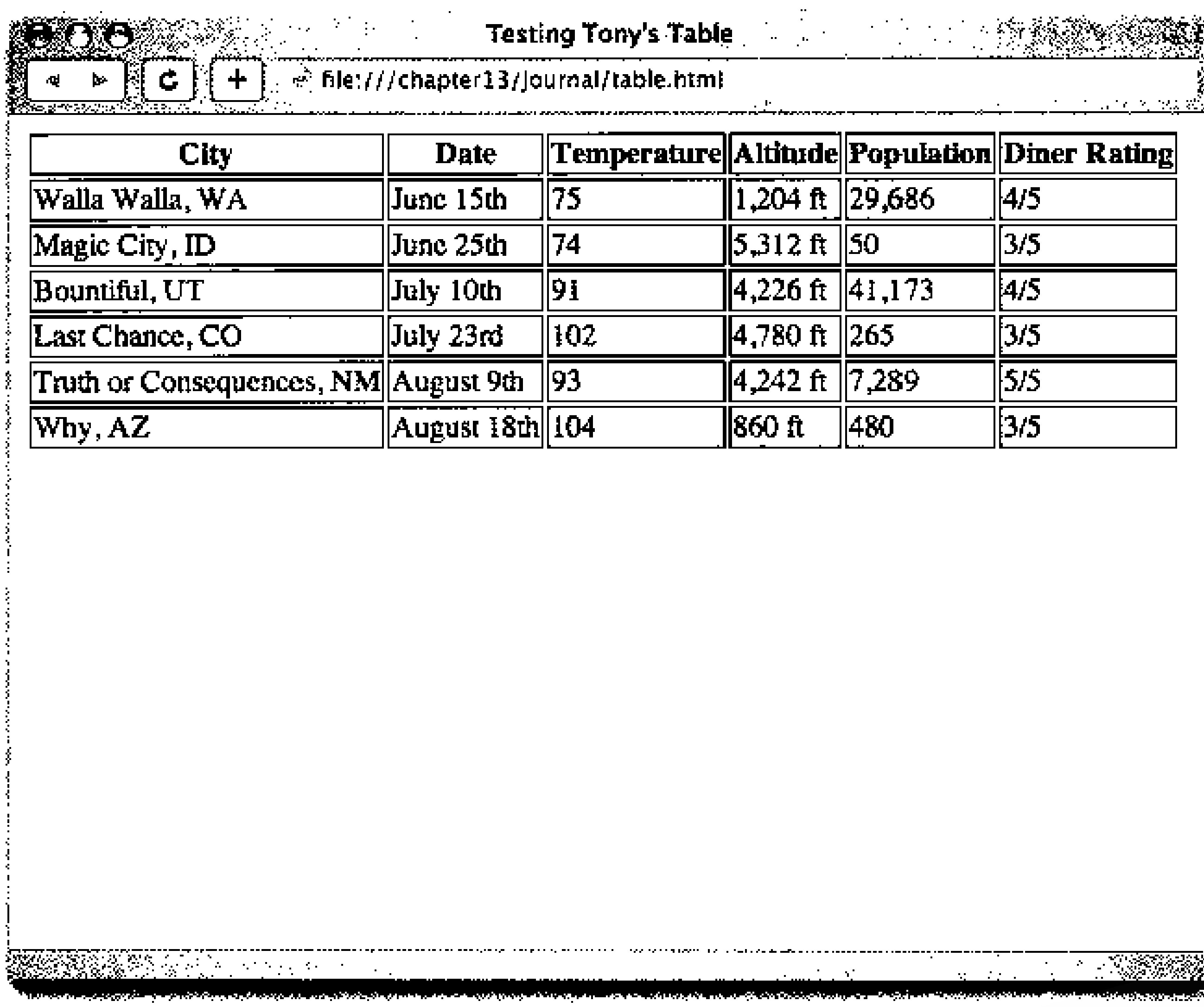
首先，输入“Testing Tony’s table”XHTML源代码。输入的过程能够帮助你了解<table>、<tr>、<th>、<td>及表头中的<td>元素的结构。输入完成后，快速测试文件，接着再添加Tony表格中剩余项目，再测试文件。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">
<head>
   <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
   <style type="text/css">
       td, th {border: 1px solid black;}
   </style>
   <title>Testing Tony's Table</title>
</head>
<body>
   <table>
      <tr>
         <th>City</th>
         <th>Date</th>
         <th>Temperature</th>
         <th>Altitude</th>
         <th>Population</th>
         <th>Diner Rating</th>
      </tr>
      <tr>
         <td>Walla Walla, WA</td>
         <td>June 15th</td>
         <td>75</td>
         <td>1,204 ft</td>
         <td>29,686</td>
         <td>4/5</td>
      </tr>
      <tr>
         <td>Magic City, ID</td>
         <td>June 25th</td>
         <td>74</td>
         <td>5,312 ft</td>
         <td>50</td>
         <td>3/5</td>
      </tr>
      <tr>
         <td>Bountiful, UT</td>
         <td>July 10th</td>
         <td>91</td>
         <td>4,226 ft</td>
         <td>41,173</td>
         <td>4/5</td>
      </tr>
      <tr>
         <td>Last Chance, CO</td>
         <td>July 23rd</td>
         <td>102</td>
         <td>4,780 ft</td>
         <td>265</td>
         <td>3/5</td>
      </tr>
      <tr>
         <td>Truth or Consequences, NM</td>
         <td>August 9th</td>
```

www.docin.com

练习答案

```
                <td>93</td>
                <td>4,242 ft</td>
                <td>7,289</td>
                <td>5/5</td>
            </tr>
            <tr>
                <td>Why, AZ</td>
                <td>August 18th</td>
                <td>104</td>
                <td>860 ft</td>
                <td>480</td>
                <td>3/5</td>
            </tr>
        </table>
    </body>
</html>
```



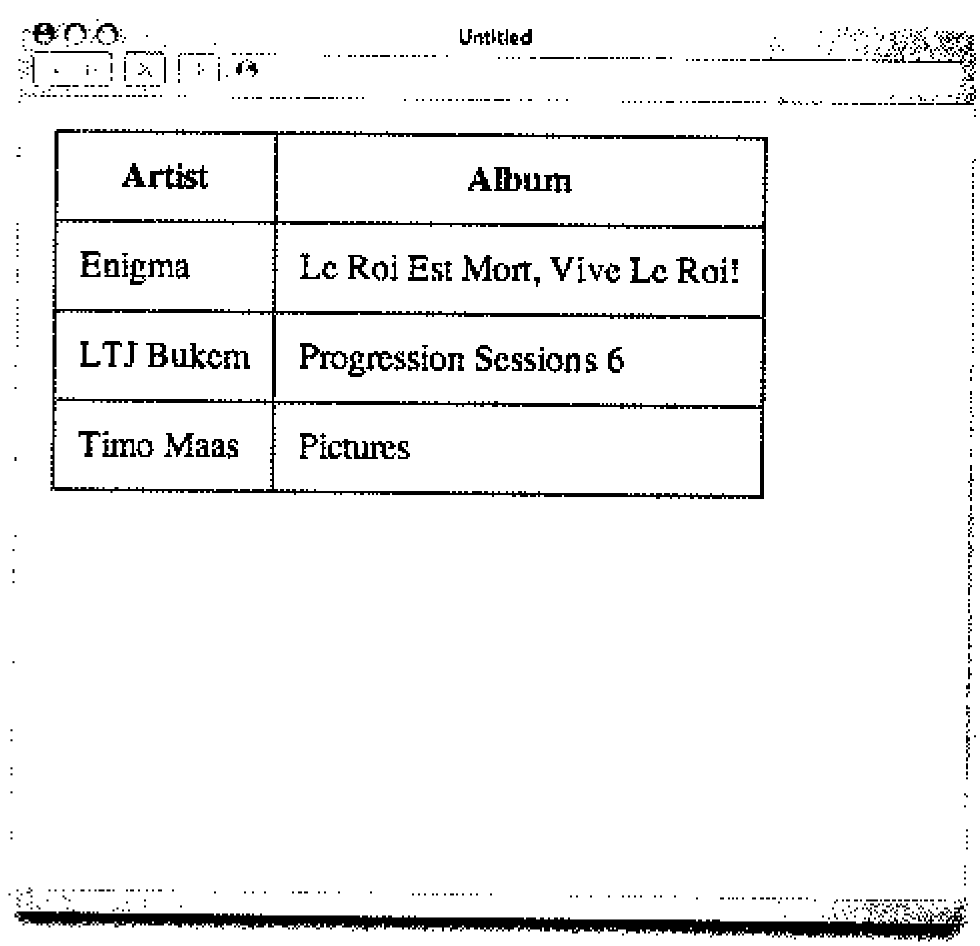
City	Date	Temperature	Altitude	Population	Diner Rating
Walla Walla, WA	June 15th	75	1,204 ft	29,686	4/5
Magic City, ID	June 25th	74	5,312 ft	50	3/5
Bountiful, UT	July 10th	91	4,226 ft	41,173	4/5
Last Chance, CO	July 23rd	102	4,780 ft	265	3/5
Truth or Consequences, NM	August 9th	93	4,242 ft	7,289	5/5
Why, AZ	August 18th	104	860 ft	480	3/5

扮演浏览器



左边是一个表格的XHTML程序。你的任务就是扮演浏览器来显示表格。这是显示的结果。

```
<table>
   <tr>
      <th>Artist</th>
      <th>Album</th>
   </tr>
   <tr>
      <td>Enigma</td>
      <td>Le Roi Est Mort, Vive Le Roi!</td>
   </tr>
   <tr>
      <td>LTJ Bukem</td>
      <td>Progression Sessions 6</td>
   </tr>
   <tr>
      <td>Timo Maas</td>
      <td>Pictures</td>
   </tr>
</table>
```

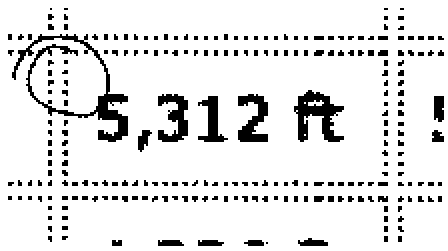


Artist	Album
Enigma	Le Roi Est Mort, Vive Le Roi!
LTJ Bukem	Progression Sessions 6
Timo Maas	Pictures

Sharpen your pencil

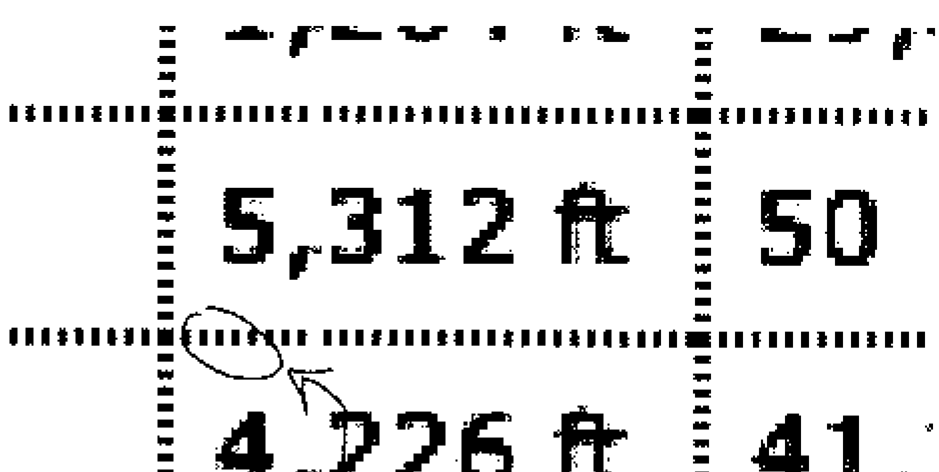
答案

双虚线让Tony的表格看上去复杂而又分散。如果每个表格单元只有一个边框的话，表格看上去会更好而且并不分散。你能够运用所学的知识想出一个办法将双虚线变为单虚线吗？



可以将border-spacing属性值设置为0以达到消除边框间距的目的。

```
table {
    margin-left: 20px;
    margin-right: 20px;
    border: thin solid black;
    caption-side: bottom;
    border-spacing: 0px;
}
```



成功了。不过仍存在两条线，只不过它们紧靠在一起罢了。那么我们得到的是双重的虚线边框。如果想要得到单元格之间的单线边框，我们该怎么办呢？

Sharpen your pencil 答案

```
.center {
    text-align: center;
}
.right {
    text-align: right;
}
```

这儿有两个类，一个类是居中对齐，一个类是右对齐。

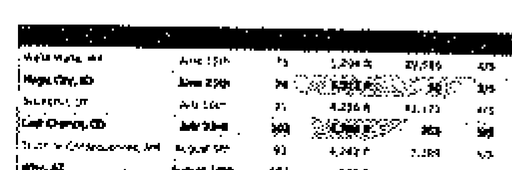
```
<table summary="This table holds data about the cities I visited on my
travels.  I've included the date I was in each city, the temperature when
I was there, and altitude and population of each city.  I've also included
a rating of the diners where I had lunch, on a scale from 1 to 5.">
  <caption>The cities I visited on my Segway'n USA travels</caption>
    <tr>
      <th>City</th>
      <th>Date</th>
      <th>Temperature</th>
      <th>Altitude</th>
      <th>Population</th>
      <th>Diner Rating</th>
    </tr>
    <tr>
      <td>Walla Walla, WA</td>
      <td class="center">June 15th</td>
      <td class="center">75</td>
      <td class="right">1,204 ft</td>
      <td class="right">29,686</td>
      <td class="center">4/5</td>
    </tr>
    <tr>
      <td>Magic City, ID</td>
      <td class="center">June 25th</td>
      <td class="center">74</td>
      <td class="right">5,312 ft</td>
      <td class="right">50</td>
      <td class="center">3/5</td>
    </tr>
    .
    .
    .
</table>
```

这儿，你只需在<td>元素中添加适当的类。

练习答案

将Magic City、Last Chance 、Why表格行设置为行行交替颜色行，你只需要在这些行中的<tr>开始标记中加入class= "cellcolor" 属性就可以了，像这样：

```
<tr class="cellcolor">
    <td>Magic City, ID</td>
    ...
</tr>
```



练习答案

连连看

确认一下你已经掌握了这些知识。那么，在每个<td>元素到对应的表格单元之间画一条带箭头的线。下面是参考答案。

```
<tr>
   <td rowspan="2">Truth or Consequences, NM</td>
   <td class="center">August 9th</td>
   <td class="center">93</td>
   <td rowspan="2" class="right">4,242 ft</td>
   <td rowspan="2" class="right">7,289</td>
   <td class="center">5/5</td>
</tr>
<tr>

   <td class="center">August 27th</td>
   <td class="center">98</td>


   <td class="center">4/5</td>
</tr>
```

Truth or Consequences, NM	August 9th	93°	4,242 ft	7,289	5/5
	August 27th	98°			4/5

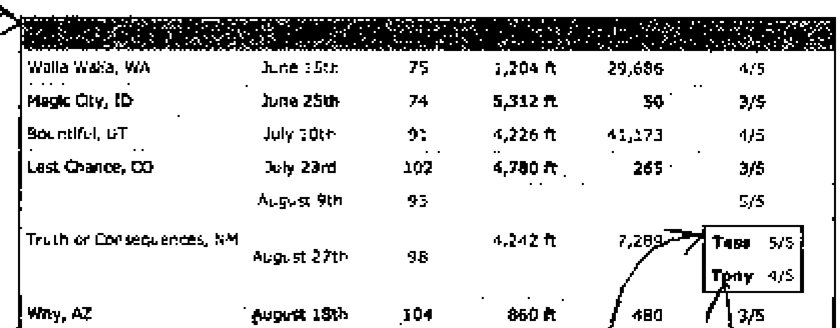
练习答案

BRAIN BARBELL

回顾一下你所练习的题目。你所要做的就是改变Tony和Tess单元格的表格表头的背景颜色，同时不能改变主表格表头的背景颜色。怎样实现它？只要使用一个只选中嵌套的表格表头的选择符就可以了。

我们可以使用子孙选择符来选中嵌套的表格表头。下面，你要做的是：

(1) 首先选择外表格……



Walla Walla, WA	June 15th	75	1,204 ft	29,686	4/5
Magic City, ID	June 25th	74	5,312 ft	50	3/5
Bountiful, UT	July 10th	91	4,226 ft	41,173	4/5
Last Chance, CO	July 23rd	102	4,780 ft	265	3/5
	August 9th	93			5/5
Truth or Consequences, NM	August 27th	98	4,242 ft	7,289	Tess 5/5 Tony 4/5
Why, AZ	August 18th	104	860 ft	480	3/5

The cities I visited on my Segway'n USA travels

(2) 其次选择内表格……

(3) 最后选择表头。

(1) (2) (3)

```
table table th {
    background-color: white;
}
```

练习
答案

[1]S	U	M	M	A	R	Y												
P																		
A				[2]H			[3]L							[4]M				[5]B
[6]N	E	S	T	E	D		I		[7]C					A				O
S				A			S		A					R		[8]W		R
				D			T		P					G		R		D
			[9]L	I	S	T	-	S	T	Y	L	E	-	I	M	A	G	E
				N			S		I					N		P		R
				G			T		O					S		P		-
				S			Y		N							I		S
							L									N		P
		[10]R					E					[11]L				G		A
	[12]B	O	R	D	E	R	-	C	O	L	L	A	P	S	E			C
		W					T					Y						I
[13]L	I	S	T	-	S	T	Y	L	E	-	P	O	S	I	[14]T	I	O	N
							P					U			O			G
			[15]M	A	R	K	E	R		[16]D	A	T	A		P			

14 xhtml表单

交互活动

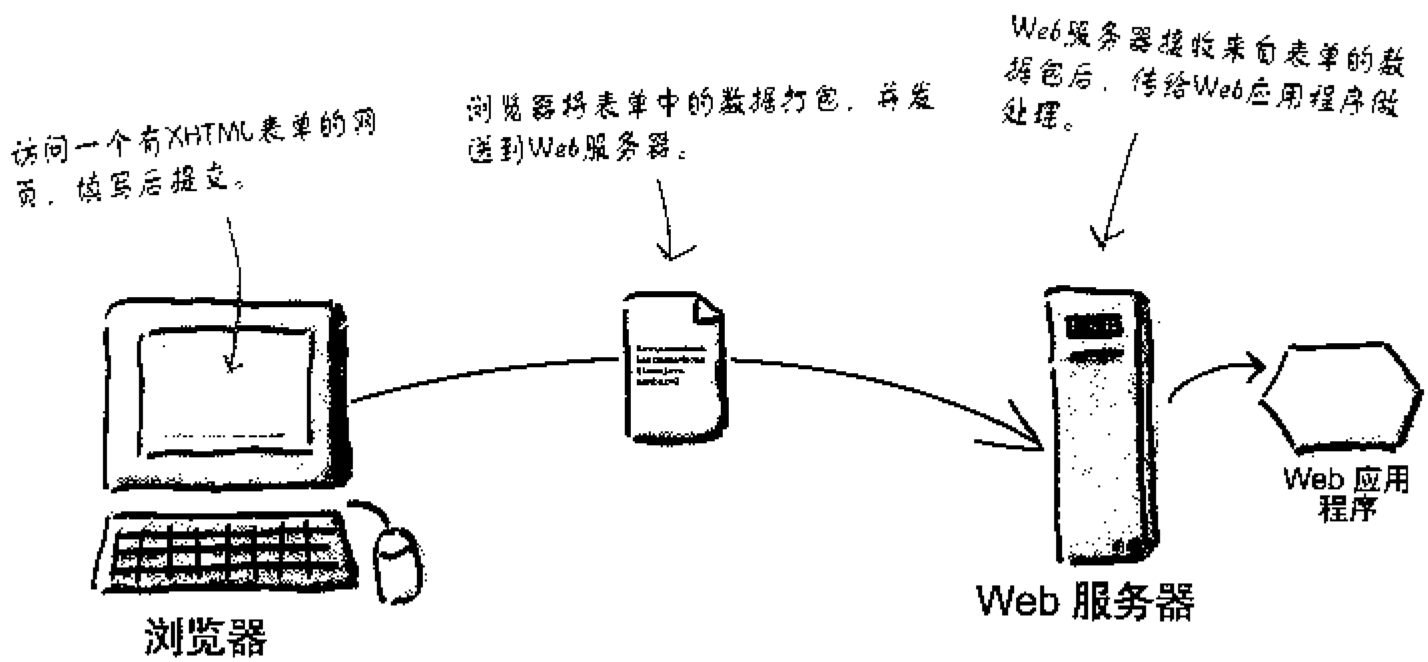


迄今为止，Web之间的通信都只是一种方式：从网页到访问者。如果访问者能够反馈，肯定非常棒，是不是？因此我们引进了XHTML表单：你一旦在网页中添加表单（同时需要Web服务器的帮助），你的网页就能够收集客户的反馈，提取订单，或者转到下一个在线游戏，或者统计热门或冷门的竞赛投票。在这一章中，你会碰到一个小队的XHTML元素，这些元素一起创建了Web表单。你还可以了解一些后台服务器处理表单的知识，此外，我们还将讨论表单的更新（一个有争论的话题，读下去会找到原因）。

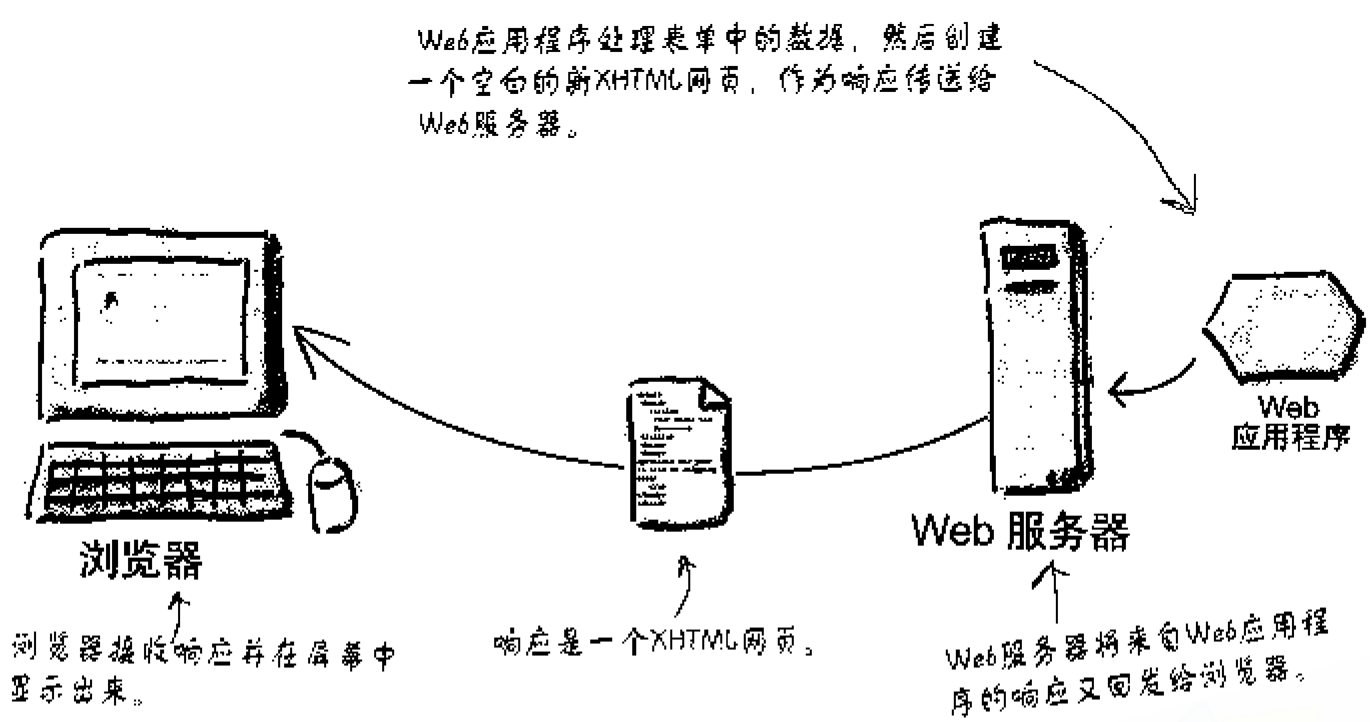
表单如何工作

如果你经常使用网络，那么你应该了解表单是什么。不过你恐怕不了解表单与XHTML之间的关系。表单基本上就是带有一块输入信息区域的网页。当提交表单时，表单中的信息被打成一个数据包发送给Web服务器，Web应用程序对之进行处理。处理完成后，你可以获得什么呢？当然是另一个响应页面了。让我们进一步了解这个过程：

1



2

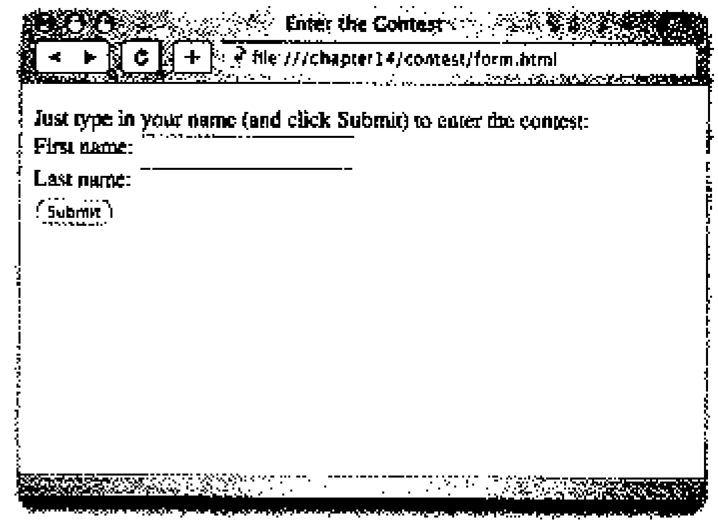


表单在浏览器中如何工作

表单只是浏览器中的一块XHTML。你会发现，可以通过添加一些新元素的方式在网页中轻而易举地创建表单。下面从浏览器的角度观察表单是如何工作的：

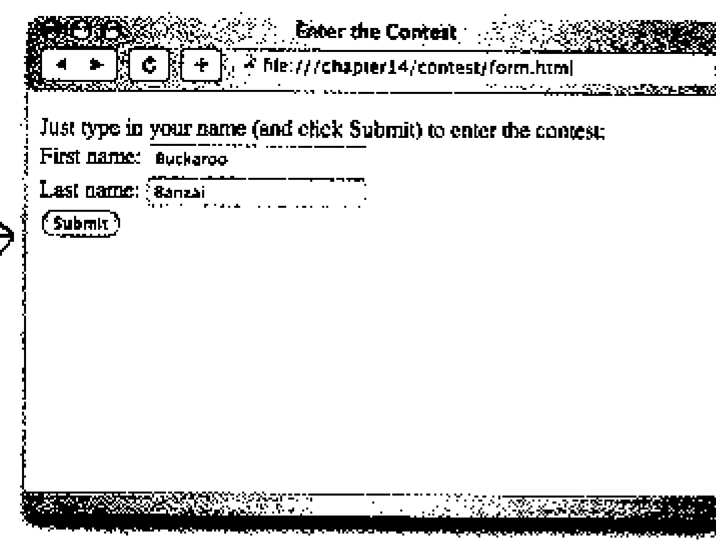
在浏览器中加载网页

浏览器像平常一样加载XHTML网页，当它碰到表单元素时，便在网页上创建一些控件，这些控件允许你输入各种各样的数据。一个控件就像是一个按钮、一个文本输入框、一个下拉菜单，总之是一些允许你输入数据的基本工具。



输入数据

使用控件输入数据。不同的控件，输入数据的方式不同。你可以在文本控件中输入一个单行的文本，或者在复选框控件中单击一个选项。稍后，我们将看一下不同种类的控件。

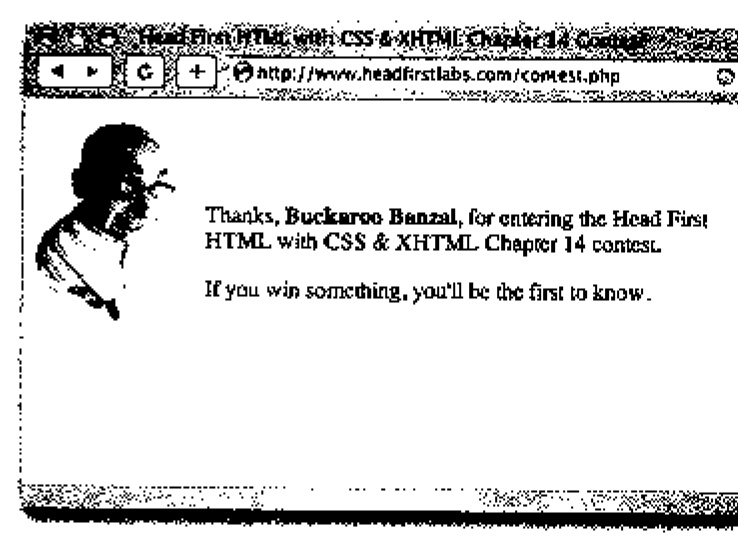


提交表单

通过单击“提交”（submit）按钮控件，来提交表单。单击按钮控件就是提示浏览器打包所有的数据，并且发送到服务器。

服务器的响应

服务器一旦收到表单中的数据，便将其传递给合适的Web应用程序来处理。处理的结果放在一个空白的新XHTML网页中，这个网页以XHTML格式返回到浏览器，然后浏览器将这个网页显示出来。



用XHTML写些什么

用XHTML创建表单，没什么深奥的。事实上，在这一章中你会看到整整一组创建表单的元素。学习表单的一个好方法就是看一点XHTML，接着试着去做。看看这个表单：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html>
   <head>
      <meta http-equiv="Content-Type"
            content="text/html; charset=ISO-8859-1" />
      <title>Enter the Contest</title>
   </head>
   <body>

     <form action="http://www.headfirstlabs.com/contest.php" method="POST">
(A)     <p>Just type in your name (and click Submit) to
           enter the contest: <br />

(B)        First name: <input type="text" name="firstname" value="" /> <br />
(C)        Last name: <input type="text" name="lastname" value="" /> <br />
(D)        <input type="submit" />

        </p>
     </form>

   </body>
</html>
```

现在这些对你来说都是老掉牙的东西了。

这就是表单。

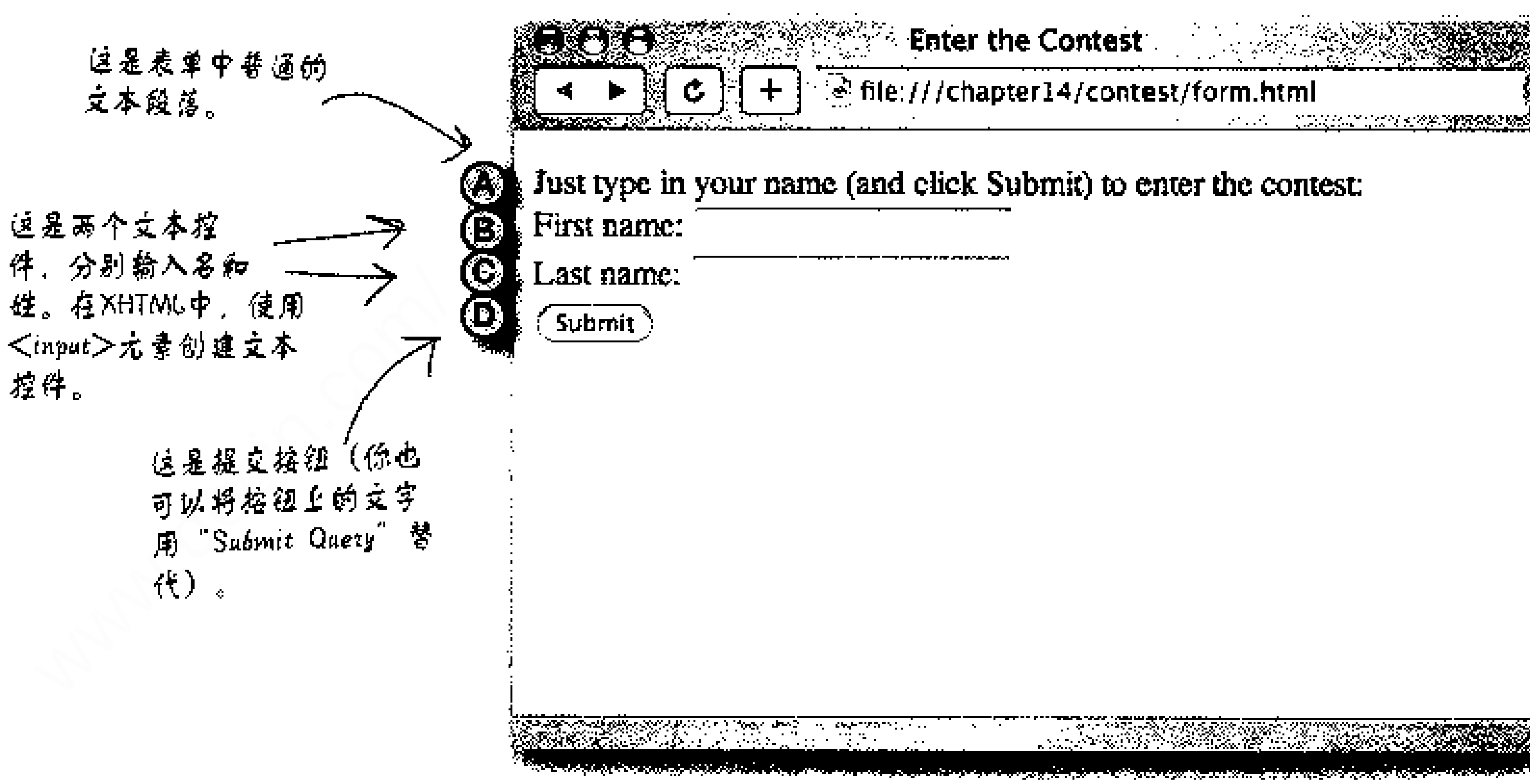
我们在本网页上获取<form>。

表单中的一堆元素。

放轻松 仔细观察表单及表单元素。这一章我们将介绍表单的各个部分。

浏览器生成了什么

大惊喜：使用**<form>**元素创建一个表单。任何元素块级别的元素都可以构成**<form>**元素，不过这里有一组专门构成表单的元素。每一个表单元素输入信息的方式都不同：例如文本框、复选框、选项菜单等。我们将一一验证这些元素。不过先回过头来看一看前面的网页。如下所示：**<form>**中的各元素在网页中是这样显示的。



练习

在"chapter/contest"文件夹中，找到contest表单，打开并加载到浏览器上，进入contest界面。

表单元素如何工作

下面进一步了解<form>元素，它不仅包括了组成表单的所有元素，而且还告诉浏览器在提交表单时将表单数据发送至何处（以及浏览器发送数据的方式）。

这是表单开始标记。所有表单元素置于开始标记中。

action属性说明了Web服务器的URL……

……这是处理表单数据的Web应用程序。

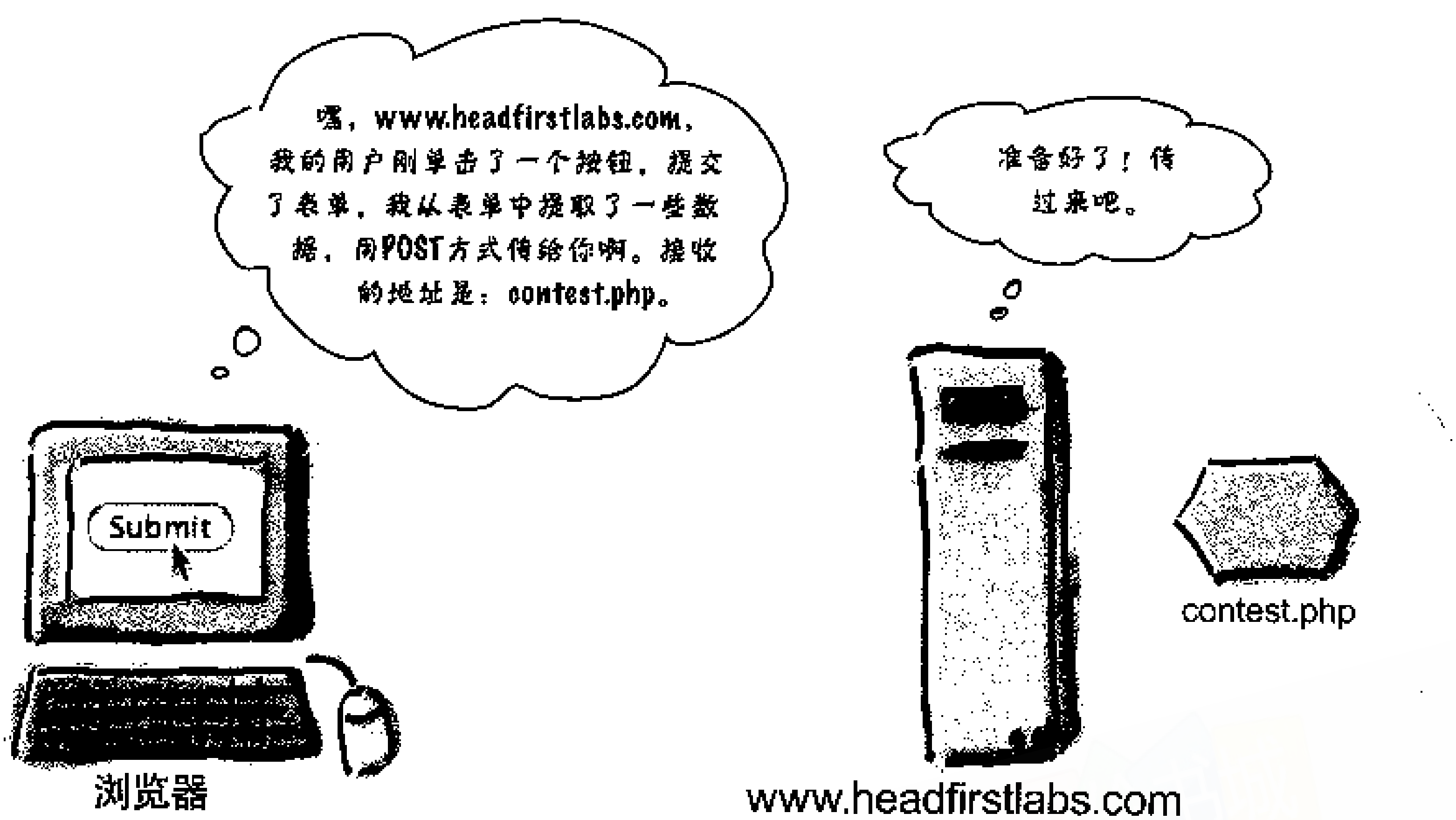
method属性指定了表单以何种方式给服务器发送数据。这里使用最普通的方式：POST。这一章的后面，我们将介绍传送数据的其他方式，还有使用或不使用POST的原因。

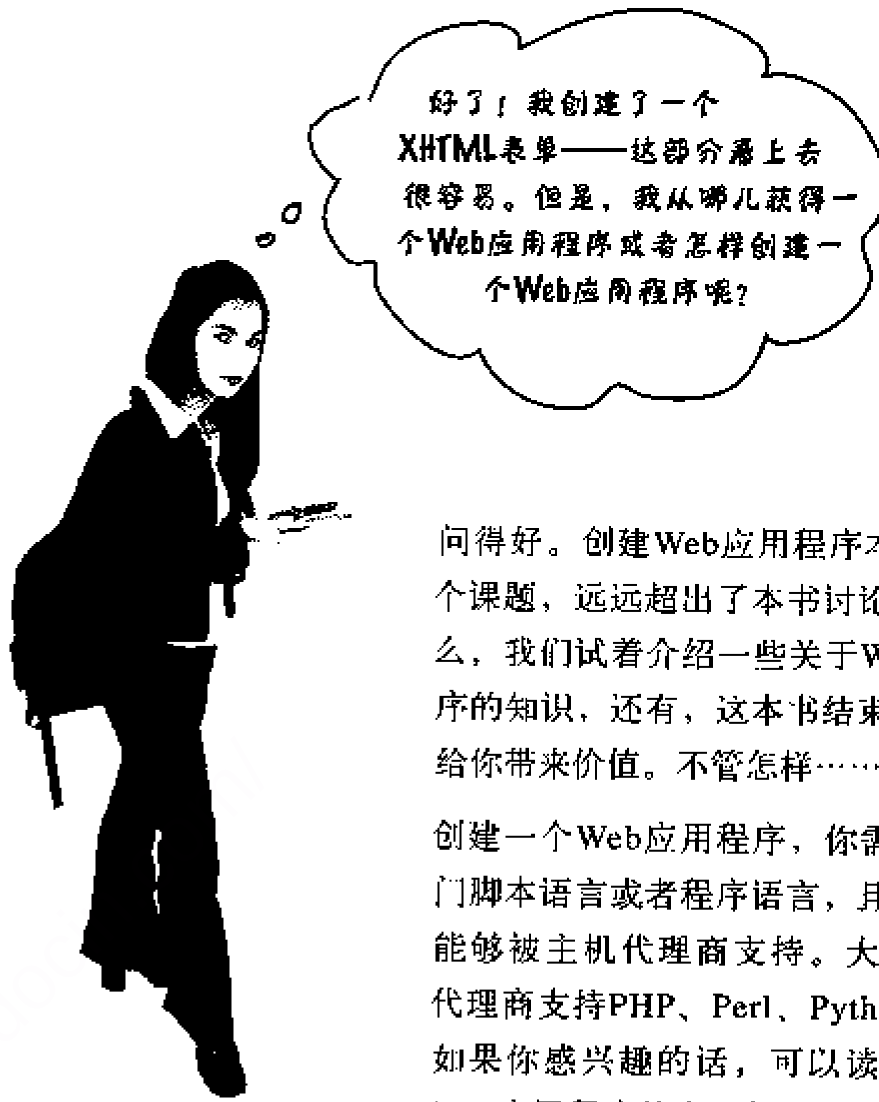
```
<form action="http://www.headfirstlabs.com/contest.php" method="POST">

    放在你表单中的所有元素。

</form>
```

接着这个结束标记结束了这个表单。





问得好。创建Web应用程序本身就是一个课题，远远超出了本书讨论范围。那么，我们试着介绍一些关于Web应用程序的知识，还有，这本书结束时绝对能给你带来价值。不管怎样……

创建一个Web应用程序，你需要掌握一门脚本语言或者程序语言，且这门语言能够被主机代理商支持。大多数主机代理商支持PHP、Perl、Python和Java，如果你感兴趣的话，可以读一本开发Web应用程序的书。检查你的主机代理商，他们有时会为用户提供一些简单的脚本，让你不用自己开发Web应用程序。

在这一章里，我们已经开发了所有需要的Web应用程序。你要做的只是将这些Web应用程序的URL放进**<form>**元素的**action**属性里。

表单中可以添加什么

你能将任意的块元素放入表单中，不过这不是我们现在该关心的。我们感兴趣的是那些创建浏览器控件的表单元素。下面是常用表单元素的一览表。先看看<input>表单元素，它在表单中扮演着重要的角色。

输入文本框

<input>输入文本框元素用来输入一行文本。它的可选属性允许你设置输入的最大字符数及文本框的大小。



type属性值为"text"的<input>元素在浏览器网页中创建了一行文本框。

大多数的表单元素都需要提供一个供Web应用程序使用的名字，我们将会看到它的一些工作情况。

使用type属性指明你想输入的是"text"。

```
<input type="text" name="fullname" />
```

<input>元素是空元素，所以使用/>符号结束标记。

注意到这两个元素都使用相同的XHTML元素，不同的只是type属性的值。

提交按钮

<input>提交元素创建了一个表单的提交按钮。当你单击这个按钮时，浏览器便发送表单至Web应用程序进行处理。



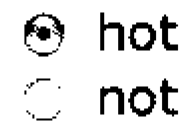
按钮上默认的字符为"Submit"（或者是"Submit Query"）。你也可以改变按钮上的文字（后边我们会讲到）。

提交按钮就是将<input>的type属性值设置为"submit"。

```
<input type="submit" />
```

输入单选框

<input>的单选框元素创建了有多个按钮的单选控件，任何时候只能选择其中的一个。就好像过去的汽车收音机按钮，按下一个按钮就会弹起其他按钮。



单选框控件只能选择一项。

每个选项置一个带radio属性的<input>标记。

单选框的按钮对应给定的选项，按钮选项名必须相同……

……不过，每个选项的值不相同。

```
<input type="radio" name="hotornot" value="hot" />
<input type="radio" name="hotornot" value="not" />
```

同样使用<input>元素，不同的只是它的type值。

输入复选框

<input>的复选框元素创建了一个复选框控件，可以被选中或不被选中。你可以使用多个复选框，根据你的爱好选中或多或少的选项。



不同于单选框，复选框允许选中零个或多个选项。

与单选框相同，每个选项置一个带复选框的<input>标签。

相关的复选框名字相同。

每个复选框的值都不相同。

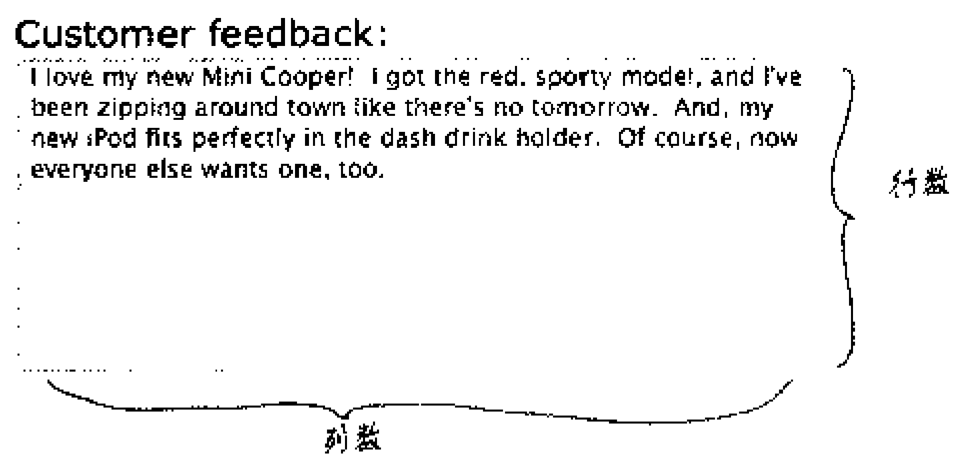
```
<input type="checkbox" name="spice" value="salt" />
<input type="checkbox" name="spice" value="pepper" />
<input type="checkbox" name="spice" value="garlic" />
```

表单中可以添加什么？（第二部分）

不是每个表单元素都要使用**<input>**。有些元素，比如菜单的**<select>**和可输入多行文本的**<textarea>**，它们都不使用**<input>**标签。那么，你为什么不先熟悉这些元素呢？顺便说一句，掌握了**<select>**和**<textarea>**之后，你差不多就掌握了90%的表单元素（和99%的常用的表单元素）。

文本区

<textarea>元素创建了一个可输入多行文本的文本框。如果输入的文本超出文本区的大小，那么文本区的右边会出现一个滚动条。



<textarea>不是一个空元素，所以它有开始和结束标记。

使用name属性获取一个唯一的名字。

属性cols告诉浏览器文本区的宽度。

```
<textarea name="comments" rows="10" cols="48"></textarea>
```

属性rows告诉浏览器文本区的高度。

任何位于<textarea>开始标签和</textarea>结束标记之间的文本都是浏览器中文本区控件的初始文本。

选择列表

<select>元素为网页创建菜单控件。菜单提供了一种在一组选项中选择的方式。<select>元素和下面的<option>元素共同创建了菜单。



选择列表元素创建的菜单如上所示（显示的效果取决于所用的浏览器）。

<select>元素作用于全部的菜单选项，并将这些选项归入一个菜单。

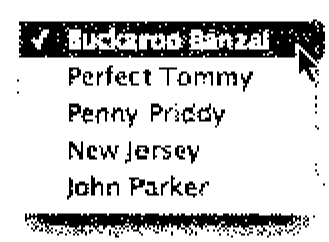
类似于其他表单元素，<select>元素使用属性name指定一个唯一的名字。

```
<select name="characters">
    <option value="Buckaroo">Buckaroo Banzai</option>
    <option value="Tommy">Perfect Tommy</option>
    <option value="Penny">Penny Priddy</option>
    <option value="Jersey">New Jersey</option>
    <option value="John">John Parker</option>
</select>
```

选项

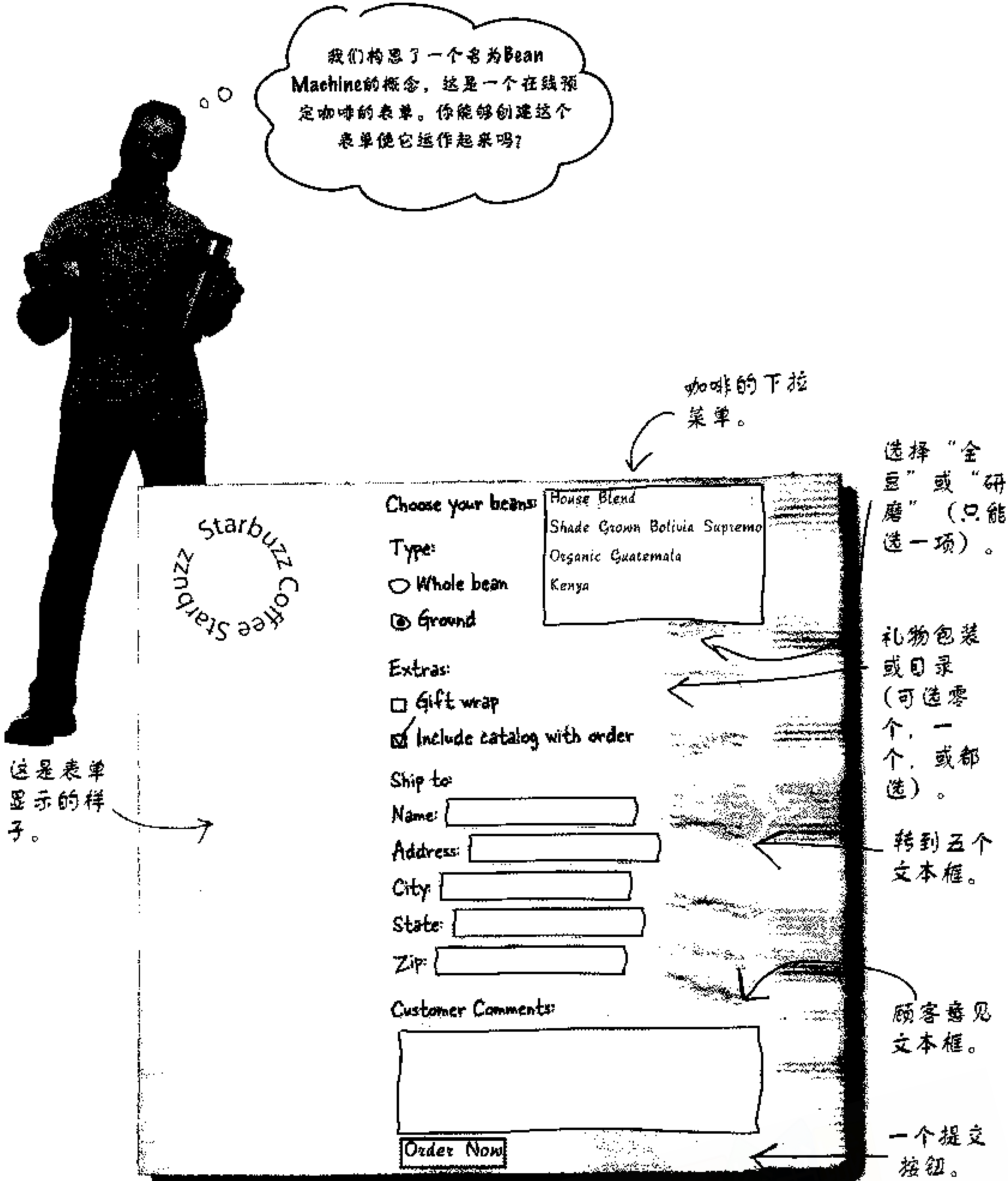
<option>元素与<select>元素共同创建一个菜单。<option>元素定义每个菜单选项。

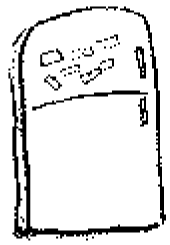
点击菜单后，菜单选项将一一罗列出来。




```
<select name="characters">
    <option value="Buckaroo">Buckaroo Banzai</option>
    <option value="Tommy">Perfect Tommy</option>
    <option value="Penny">Penny Priddy</option>
    <option value="Jersey">New Jersey</option>
    <option value="John">John Parker</option>
</select>
```

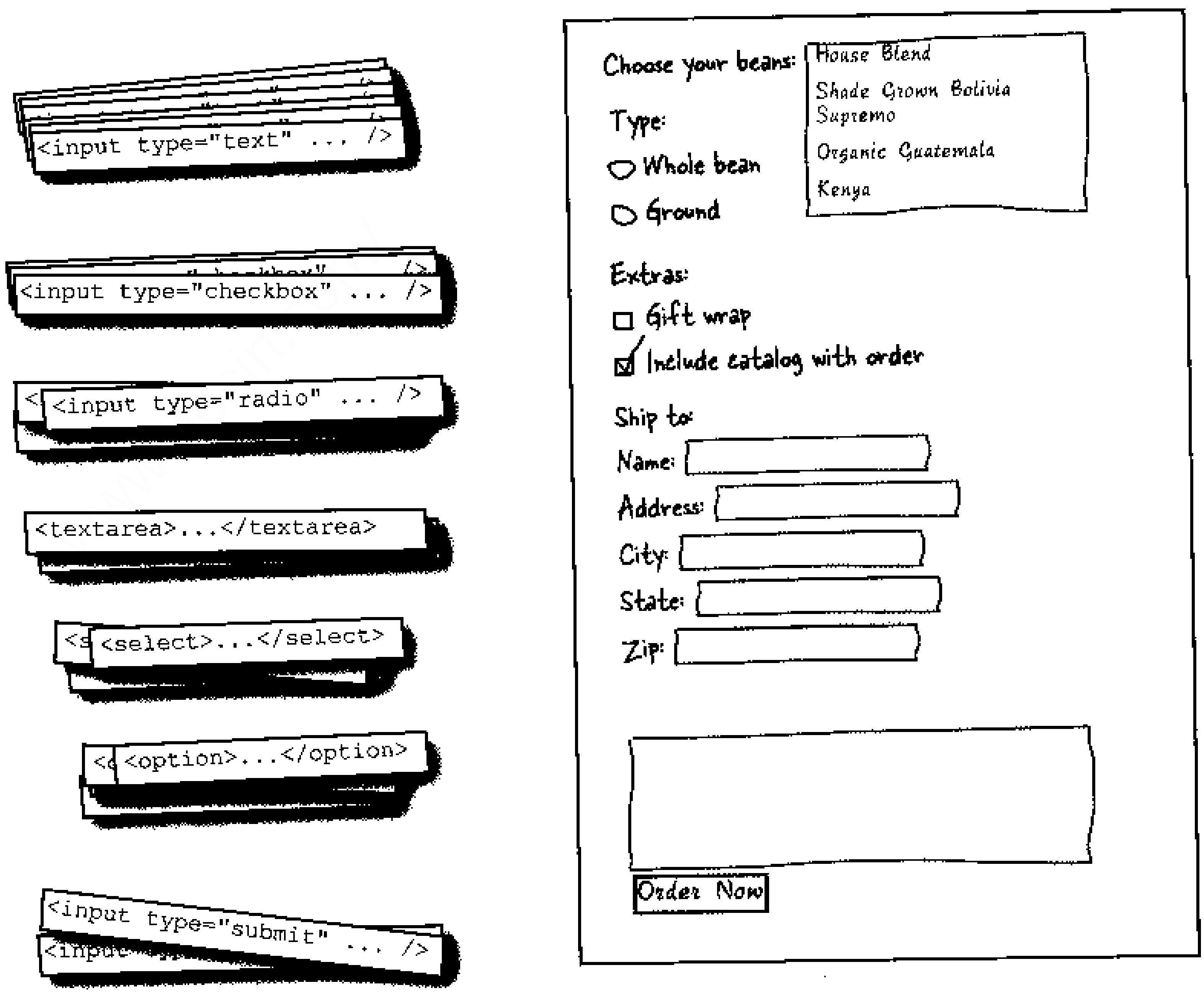
<option>元素内容描述了菜单选项，每个菜单选项包含了一个提交的菜单选项值。


我们构思了一个名为Bean Machine的概念，这是一个在线预定咖啡的表单。你能够创建这个表单使它运作起来吗？
咖啡的下拉菜单。
选择“全豆”或“研磨”（只能选一项）。
礼物包装或目录（可选零个，一个，或都选）。
转到五个文本框。
顾客意见文本框。
一个提交按钮。
这是表单显示的样子。
Starbuzz Coffee Starbuzz
Choose your beans:
House Blend
Shade Grown Bolivia Supremo
Organic Guatemala
Kenya
Type:
Whole bean
Ground
Extras:
Gift wrap
Include catalog with order
Ship to:
Name:
Address:
City:
State:
Zip:
Customer Comments:
Order Now



标记帖

你的任务是提取表单元素，并把它们放到图中相应的控件上方。你不需要使用下面所有的标记帖就可以完成这个任务，左边已经给出一些表单元素。进一步学习之前，核对一下本章后面的答案。



准备创建Bean Machine表单

在创建表单之前，打开“chapter14/starbuzz”文件夹，找到“form.html”文件。打开它，仔细通读一遍，这些都是XHTML的基础。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
                      "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en" >
   <head >
      <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
      <title>The Starbuzz Bean Machine</title>
   </head>
   <body>

      <h1>The Starbuzz Bean Machine</h1>
      <h2>Fill out the form below and click submit to order</h2>


   </body>
</html>
```

到这里为止都是标识网页的标题，以及操作该网页的指示。

表单放在这儿。

现在我们将创建这些表单，暂时不采用任何已使用在Starbuzz网站上的样式。那样的话，我们可以将精力集中在XHTML表单上，稍后，我们会加入这些样式。

搞清将什么放入表单元素

该是添加**<form>**元素的时候了。创建**<form>**元素时，首先应该知道的是将处理你的表单数据的Web应用程序的URL。我们已经为你介绍过了。下面是处理Starbuzz订单的Web应用程序：

http://www.starbuzzcoffee.com/processorder.php

这个URL指向Starbuzz Coffee网站……

……并发送至服务器上的processorder.php Web应用程序。这个应用程序知道怎样提取我们创建的表单订单。

添加表单元素

一旦你了解了处理表单的Web应用程序的URL，你要做的就是将它插入到表单元素的action属性中，就像这样（依照下面的内容，将改动输入到你的XHTML文件中）：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
                      "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en" >
   <head>
      <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
      <title>The Starbuzz Bean Machine</title>
   </head>
   <body>

      <h1>The Starbuzz Bean Machine</h1>
      <h2>Fill out the form below and click submit to order</h2>

      <form action="http://www.starbuzzcoffee.com/processorder.php" method="POST">

      </form>
   </body>
</html>
```

这是表单元素。

这个action属性包含了Web应用程序的URL。

记住：我们使用POST方法将表单数据发送到服务器。后面将会介绍更多这方面的内容。

继续添加表单结束标记。

到目前为止，你已经做得很好了。不过一个空的<form>元素没有多大的用处。回过头看一下表单的草图，图上还有很多东西需要添加。我们就从简单的“ship to:”开始，它包含了一组文本输入。你已经了解了一些关于文本输入的知识了，现在，我们需要更加深入地了解。下面，我们介绍Starbuzz表单的文本输入：

```
<input type="text" name="name" />
<input type="text" name="address" />
<input type="text" name="city" />
<input type="text" name="state" />
<input type="text" name="zip" />
```

我们使用<input>元素添加一些不同的控件。type属性决定了控件的类型。

表单中的每个输入区分别对应一个文本输入：Name、Address、City、State、Zip。

此处type的类型为“text”，因为需要一个文本输入控件。

Name属性是用户输入数据的标识符。注意：每个name设置为不同的值。让我们看看它是怎么起作用的。

表单元素如何工作

下面介绍**name**属性：它在表单和处理表单的Web应用程序之间建立连接，如下所示：

表单中的每个输入控件都有一个name属性。

当你把一个表单元素输入XHTML文件时，赋予它一个唯一的名字。在文本输入中你会看到：

```
<input type="text" name="name" />
<input type="text" name="address" />
<input type="text" name="city"  />
<input type="text" name="state"  />
<input type="text" name="zip"  />
```

注意：此处我们设置了一个名为"name"的名字（这种格式很好）。

每个<input>元素都有自己的名字。

当提交一个表单时，浏览器就会使用唯一的名字来打包所有的数据：

那么在表单中输入Name、Address、City、State、Zip，单击提交按钮。浏览器将你的每个数据和标签赋予唯一的名字属性值。接着，浏览器将这些名字及它们的值发送给服务器。就像这样：

这是输入表单的内容。

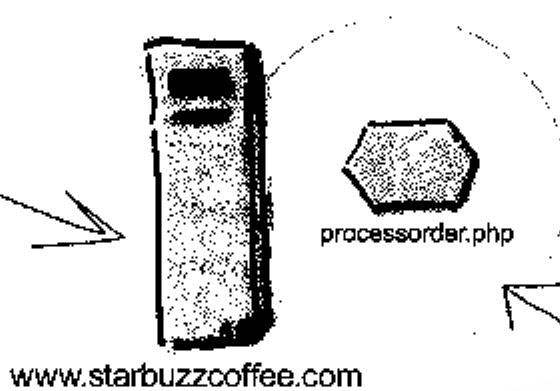
Name: Buckaroo Banzai
Address: Banzai Institute
City: Los Angeles
State: CA
Zip: 90050

表单的每个元素有一个唯一的名字。

```
name = Buckaroo Banzai
address = Banzai Institute
city = Los Angeles
state = CA
zip = 90050
```

每个唯一的名字从输入到表单的数据中获得一个值。

浏览器为服务器打包的数据。



Web应用程序需要有标签的表单数据，这样才能将数据区分开。

there are no Dumb Questions

问：<input>文本和<textarea>有什么区别？

答： 输入单行文本使用的是<input>，例如一个名字或者一个邮编；而输入多行文本使用的是<textarea>。

问：我可以在提交按钮上标上其他的字符，而不是"Submit"吗？

答： 当然可以。在这个元素中添加一个value属性，然后给它设置一个值，比如"Order Now"。你也可以使用文本输入的value属性设置一个默认的文本。

问：对于<input>文本和<textarea>，输入文字的多少有限制吗？

答： 不管是<input>文本还是<textarea>，浏览器都限制了你可以输入的文本字数。不过，通常限制的字数比你要输入的字数要多。如果你需要限制用户输入到<input>文本的字数，你可以使用maxlength属性设置其值为一个特定的数字。例如：maxlength="100"限制用户最多能输入100个字符。不过，对于<textarea>，XHTML中没有限制用户输入字数的方法。

问：我还是不明白名字是怎样匹配表单数据的。

答： 你知道每一个表单元素有一个唯一的名字，你也知道每一个元素有一个相应的值。当单击提交按钮时，浏览器提取所有的名字连同它们的值一起发送给服务器。例如，当你将邮编"90050"输入到名为"zip"的<input>文本元素中时，提交表单后浏览器就会发送"zip=90050"给服务器。

问：Web应用程序如何知道我在表单中所使用的名字？换句话说，我该怎样挑选表单元素的名字？

答： 问得好。Web应用程序还有其他工作方式：你必须知道Web应用程序接受的是哪些表单名字，然后你所写的表单能够匹配这些表单名字。如果你使用了一个别人写的Web应用程序，他们会告诉你该使用哪些名字，或者给你提供应用程序的文档说明。一个好方法就是向主机代理商寻求帮助。

问：为什么<option>元素没有name属性？而表单的其他元素都有。

答： 问得好。全部的<option>元素事实上都是菜单的一部分，都是由<select>元素创建的。所以，我们只需要一个名字命名整个菜单就可以了，这在<select>元素中就已经定义过了。换句话说，因为<select>元素已经定义了整个菜单的名字，那么<option>元素就不需要name属性了。记住，当提交表单时，只把当时选中的选项值连同这个名字发送给服务器。

问：你不是说过表单的每个元素的名字是唯一的吗？但是，所有的<input>单选框元素的名字怎么都是相同的呢？

答： 是的。单选框按钮一般是一组。考虑一下：如果你按下一个按钮，其余的就会弹起。所以，如果你使用相同的名字，浏览器就可以知道一组单选框按钮。比如说，你有一系列值为"red"、"green"、"blue"而名字为"color"的单选框按钮。这些按钮名都是color，不过某一时间只能选中一个按钮，所以这一组的按钮命名为一个名字才有意义。

问：那复选框呢？它的工作原理跟单选框按钮是一样的？

答： 是的。唯一不同的是，复选框允许选中多个选项。

当浏览器将表单数据发送给服务器时，浏览器将所有复选框的值合并为一个值连同复选框的名字一起发送。比如，一个有"salt"、"pepper"、"garlic"选项的"spice"复选框，全部选中后，浏览器就会发送"spice=salt&pepper&garlic"给服务器。

问：我真的需要了解数据发送给浏览器过程的所有细节吗？

答： 需要知道的是你使用的Web应用程序所期望的表单元素的名字和类型是什么。除此之外有帮助的是知道Web应用程序是如何工作的。关于发送至服务器的后台细节部分你不需要知道。

将<input>元素加到XHTML中

现在,我们已经将那些**<input>**元素加到表单中了。下面检查增添的程序,无错后将改动保存到“form.html”文件中。

这些仅仅是“form.html”文件的片断。我们要在这儿保存一些树。

你只能将块元素直接嵌入表单。

首先,将所有元素嵌入<p>元素中。

```
<form action="http://www.starbuzzcoffee.com/processorder.php" method="POST">
  <p>Ship to: <br />
      Name: <input type="text" name="name" /> <br />
      Address: <input type="text" name="address" /> <br />
      City: <input type="text" name="city" /> <br />
      State: <input type="text" name="state" /> <br />
      Zip: <input type="text" name="zip" /> <br />
  </p>
  <p>
      <input type="submit" value="Order Now" />
  </p>
</form>
```

这些都是<input>元素:每个都是表单“Ship to”部分的文本输入框。

对每个输入框,我们都附加了标签,以便使用户了解该在文本输入框中填入些什么内容。

还有,你应该知道<input>是内联元素,那么如果你要在<input>元素之间添加换行的话,就得插入一个
。那就是为什么你需要在一个段落中嵌入这些个
的原因了。

最后,不要忘记用户需要一个提交按钮来提交表单。所以,加入一个提交按钮,方法是在程序末尾插入一个类型为“submit”的<input>元素。同时,将此元素的值设置为“Order Now”,这样就能将按钮上的文本内容“Submit”改成“Order Now”了。

完成这些改动后,将它们保存到文件“form.html”中并运行一下。

不要忘记校验你的HTML程序。表单元素同样需要校验。

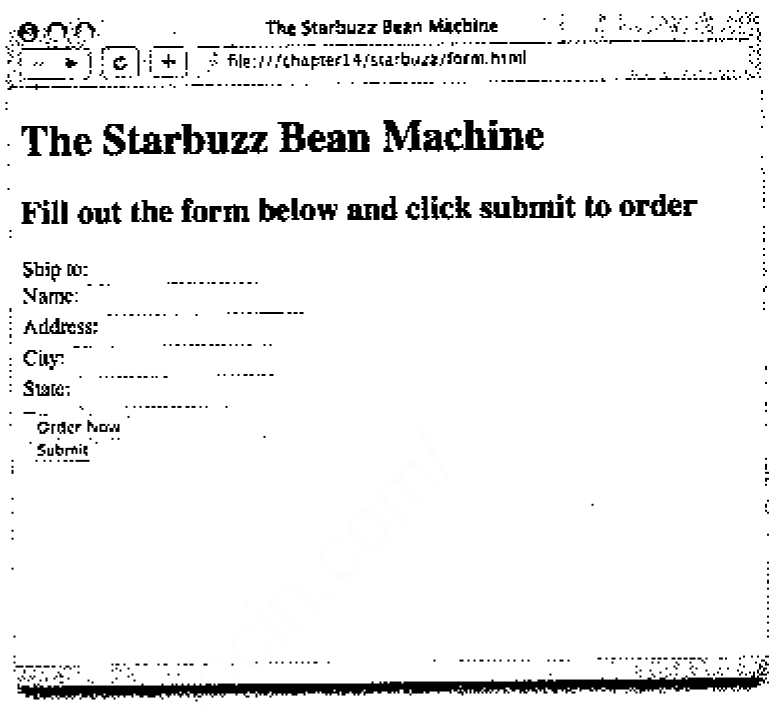
form-al测试

重新加载页面，在文本输入框中输入信息，然后提交表单。当你提交表单后，浏览器将这些数据打成数据包发送到`action`属性指定的URL中，地址为www.starbuzzcoffee.com。

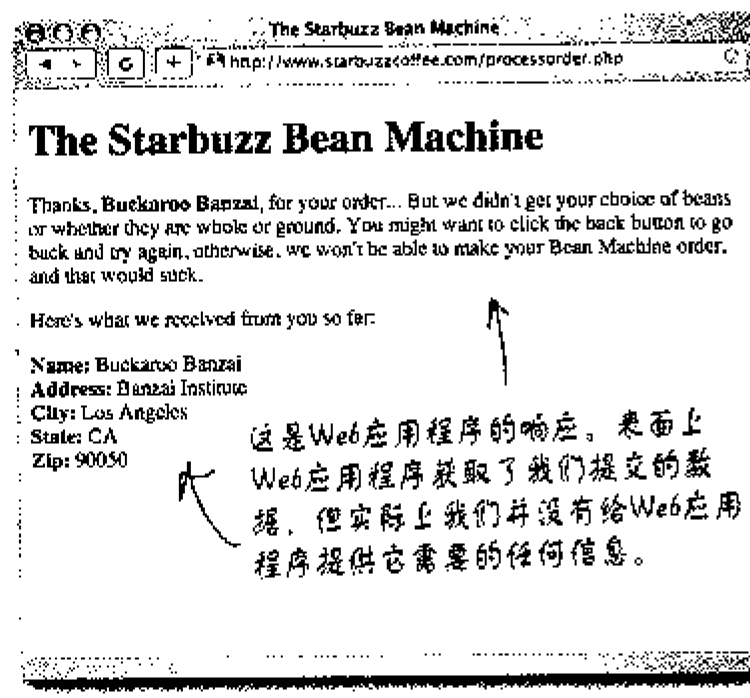
你认为我们给你的是一个能真正运行的小例子，对吗？严格地说，starbuzzcoffee.com已经准备好接收你的表单了。开始吧。

提交表单后，注意地址栏（address bar）中URL的变化（URL位于地址栏的action属性中）。

这是表单。



这就是提交表单后的响应。



这是Web应用程序的响应。表面上Web应用程序获取了我们提交的数据，但实际上我们并没有给Web应用程序提供它需要的任何信息。

在表单中添加更多的<input>元素

看样子，不告诉Web应用程序我们想要的咖啡豆和咖啡豆类型（研磨或全豆），它就不让我们继续下去。那么，首先，我们添加一个咖啡豆的选项，也就是在表单中添加一个`<select>`元素。我们记得`<select>`元素包括了一系列的选项，每个选项都是下拉菜单中的选择项。还有，每一个选项匹配一个值。当提交表单时，便将选中的菜单选项值发送给服务器。那么，翻开下一页，让我们来添加`<select>`元素。

添加<select>元素

```
<form action="http://www.starbuzzcoffee.com/processorder.php" method="POST">
   <p> Choose your beans:

      <select name="beans">

         <option value="House Blend">House Blend</option>

         <option value="Bolivia">Shade Grown Bolivia Supremo</option>

         <option value="Guatemala">Organic Guatemala</option>

         <option value="Kenya">Kenya</option>

      </select>

   </p>

  <p>Ship to: <br />
     Name: <input type="text" name="name" /> <br />
     Address: <input type="text" name="address" /> <br />
     City: <input type="text" name="city" /> <br />
     State: <input type="text" name="state" /> <br />
     Zip: <input type="text" name="zip" /> <br />
  </p>
  <p>
     <input type="submit" value="Order Now" />
  </p>
</form>
```

这就是我们添加的空白的新<select>元素。它也有一个唯一的名字。

在<select>元素里，对每个咖啡豆选项，我们都设置了一个<option>元素。

再靠近一点HTML

让我们详细了解一下<option>元素。

每个选项都有一个值。

<option>元素的内容作为下拉菜单的标签。

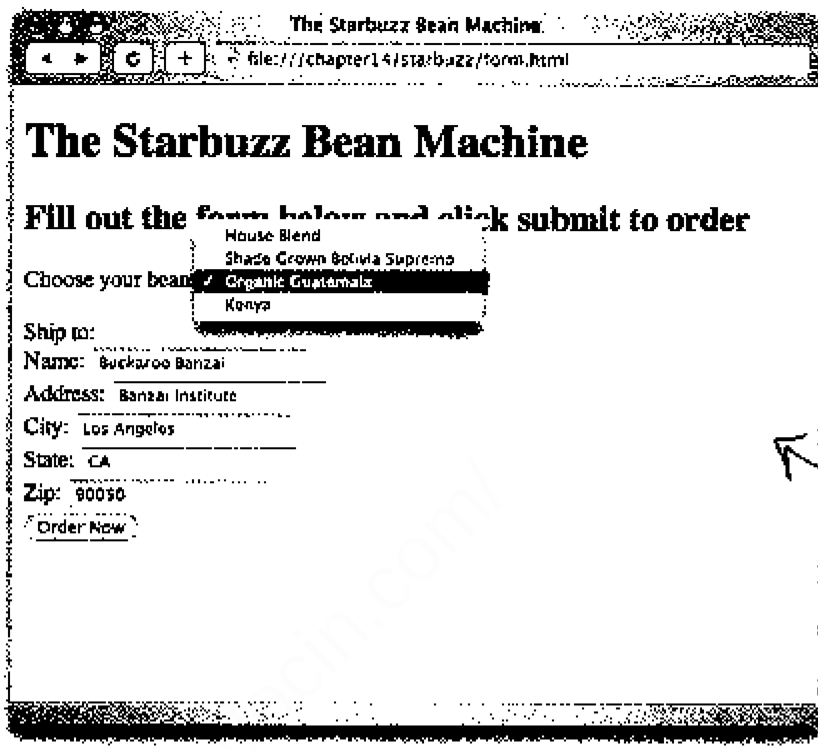
```
<option value="House Blend">House Blend</option>
```

当浏览器将表单元素的名字和值打成数据包时，它使用<select>元素的名字，以及选中的选项自身的值。

在这个例子中，浏览器发送至服务器的数据为：beans= "House Blend"。

测试运行<select>元素

让我们测试一下`<select>`元素。将`<select>`元素再次加载到你的页面上，一个全新的页面正等待着你！选择你喜欢的咖啡，填完剩余的表单文本输入框并提交你的订单。

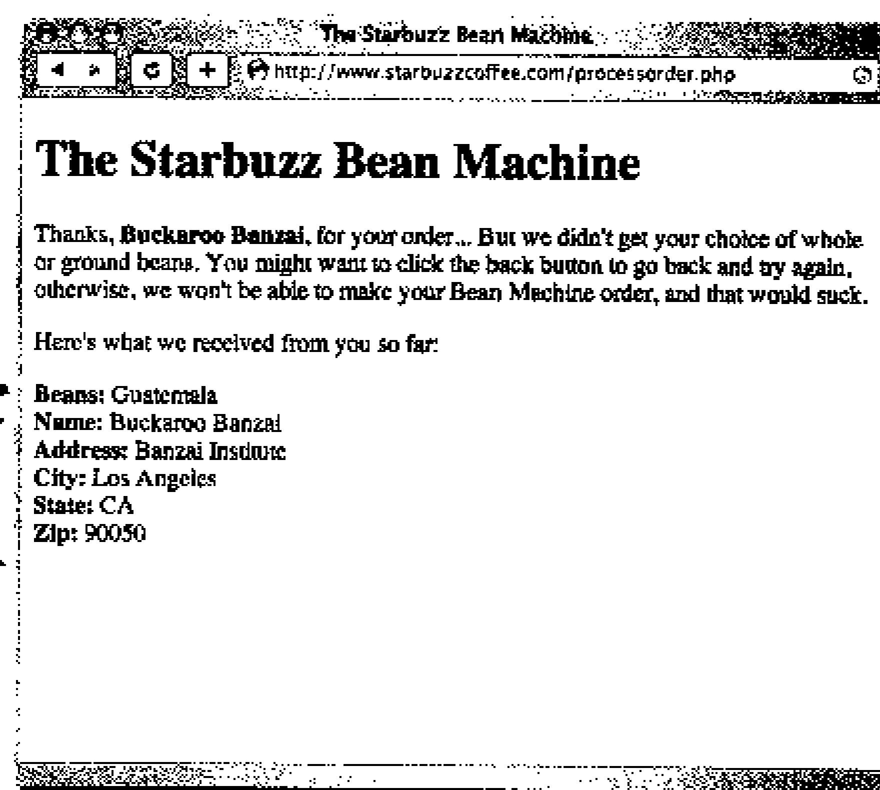


这是添加了<select>后的表单。注意：所有的选项都在表单中。

我们依旧没有给Web应用程序所需的一切。不过现在它却在表单中获得了它所需的一切。

这是<select>选择的结果。

这是所有的文本输入。



BRAIN POWER

将这个<select>元素的name属性改为“thembeans”，将改动的表单重新加载网页并重新提交你的订单。这个改动会怎样影响你从Web应用程序那里获得的结果呢？

做完这个练习后，确认将name属性改回原先的“beans”。

给顾客一个选择，是whole（全）咖啡豆呢还是ground（研磨的）咖啡豆？

顾客需要能够在他们的订单中选择咖啡豆，是whole（全）咖啡豆呢还是ground（研磨的）咖啡豆。这样的话，我们就应该使用一个单选框按钮。单选框按钮就像是老牌的汽车收音机的按钮那样——同一时间你只能按下一个按钮。它们在XHTML中运行的方式就是为每个按钮创建type为radio的**<input>**。对于咖啡豆，你就得创建两个按钮，一个为whole，一个为ground。如下所示：

这里有两个按钮：一个为whole，一个为ground。

```
<p>Type: <br />
   <input type="radio" name="beantype" value="whole"  /> Whole bean <br />
   <input type="radio" name="beantype" value="ground" /> Ground
</p>
```

使用<input>元素，设置其type为“radio”。

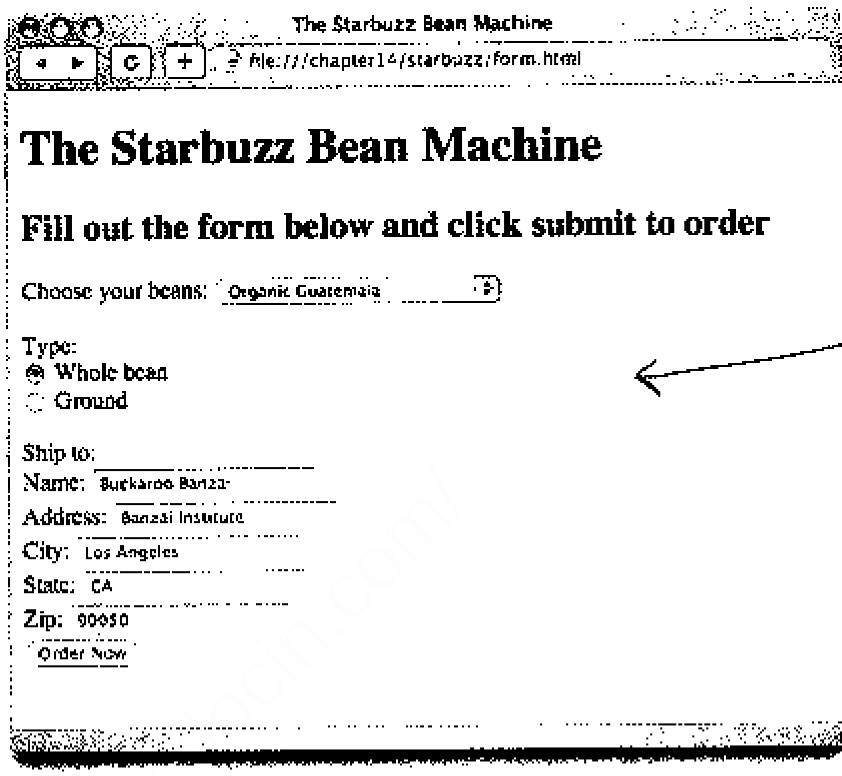
赋予唯一的name，同一组的所有单选框按钮共享一个相同的name。

这是value的值，发送给Web应用程序，只发送其中的一个值（提交表单时选中的那个值）。

注意：我们总是在元素的右手边标记单选框按钮。

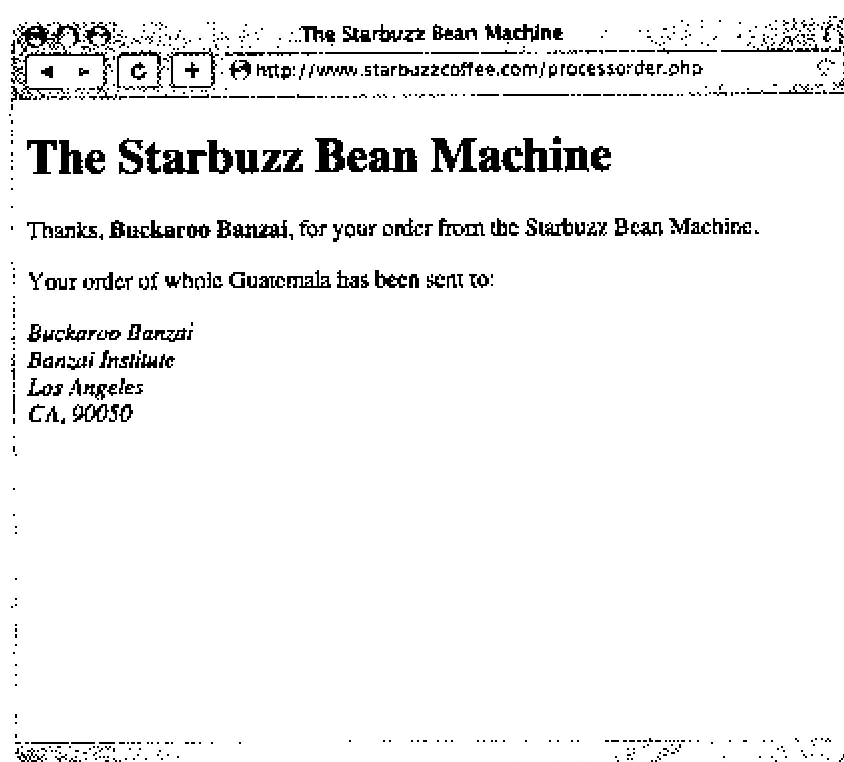
显印单选框按钮

在前面的网页中加入单选框按钮的XHTML并添加到你的XHTML程序中，放在包含`<select>`元素的段落下面。重新加载页面后，再一次提交订单。



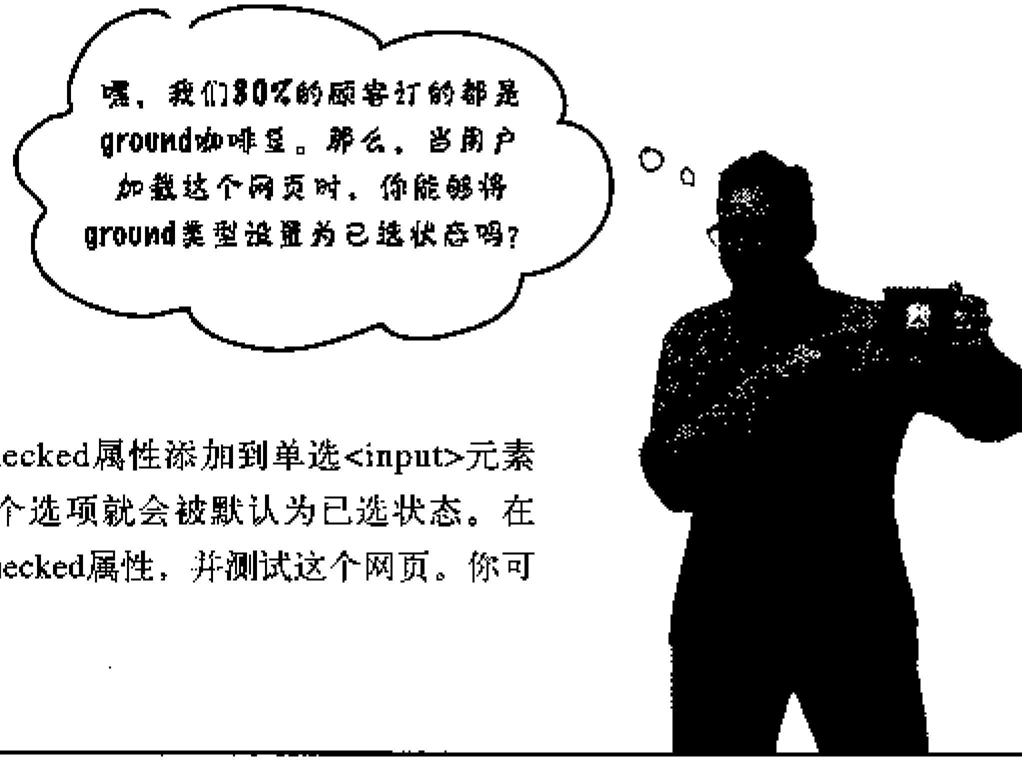
当你重新加载网页时，你会发现没有单选框按钮被按下，这取决于你使用的浏览器版本。

Starbuzz接受了我们的订单。我们甚至还没有完善订单呢。还要在表单中添加礼物选项和顾客意见区呢。



当表单中的元素不全时，订单是怎么发送出去的？这取决于Web应用程序的设计。在表单元素不全的情况下，Web应用程序设计可以处理订单，即使礼物包装（gift wrap）、商品目录（catalog）选项没有连同表单中其他元素一起提交。如果你想知道Web应用程序要求的是哪些表单元素，唯一的方法就是询问Web应用程序的开发者，或者查看Web应用程序的文档说明。

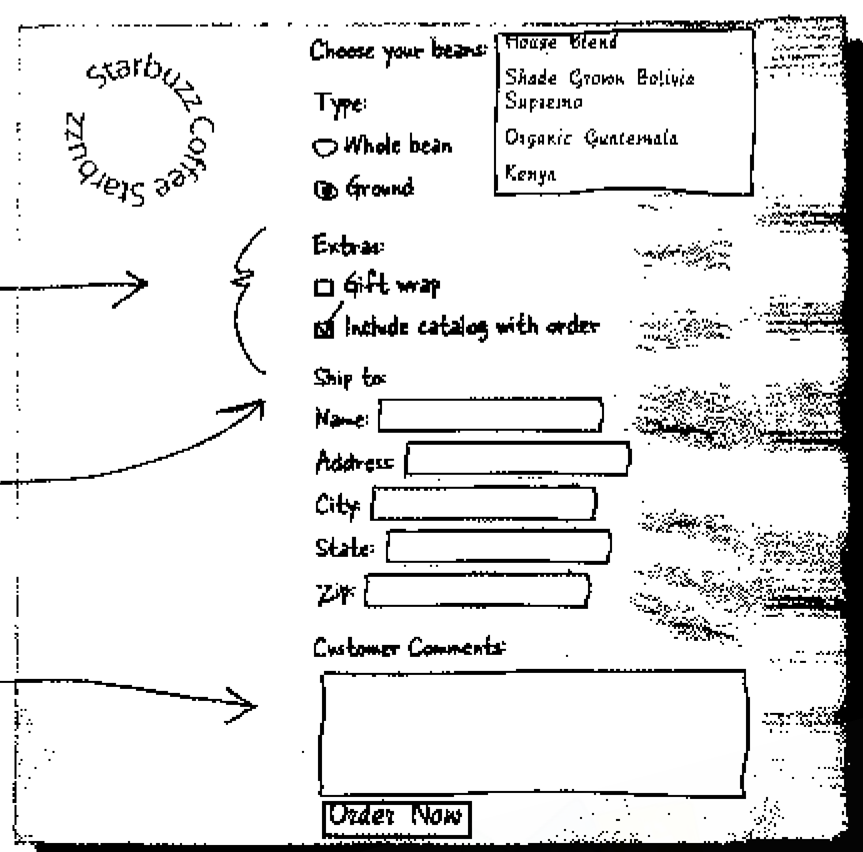
练习



如果你将一个值为“checked”的checked属性添加到单选<input>元素中，那么浏览器在显示表单时，那个选项就会被默认为已选状态。在ground 的单选<input>元素中添加checked属性，并测试这个网页。你可以在本章的后面找到答案。

完善表单

已经差不多了。你仅仅还有两个部分需要添加到表单中：一部分是带两个复选框的“Extras”；一部分是顾客意见。因为你已经掌握了表单的诀窍（hang），所以我们可以加快速度同时添加这两部分。



“Extras”部分包括了两个复选框，一个是礼物包装；另一个包含了一个商品目录。

从图上看，“include catalog”选项应该是默认选中。

顾客意见部分是一个<textarea>。

添加复选框和文本区

检查新的XHTML,将它添加到你的“form.html”文件中。

```
<form action=“http://www.starbuzzcoffee.com/processorder.php” method=“POST”>
   <p> Choose your beans:

      <select name=“beans”>

         <option value=“House Blend”>House Blend</option>

         <option value=“Bolivia”>Shade Grown Bolivia Supremo</option>

         <option value=“Guatemala”>Organic Guatemala</option>

         <option value=“Kenya”>Kenya</option>

      </select>

   </p>

   <p>Type: <br />

      <input type=“radio” name=“beantype” value=“whole” /> Whole bean <br />
      <input type=“radio” name=“beantype” value=“ground” checked=“checked” /> Ground

   </p>

   <p>Extras: <br />
      <input type=“checkbox” name=“extras[]” value=“giftwrap” /> Gift wrap <br />
      <input type=“checkbox” name=“extras[]” value=“catalog” checked=“checked” /> Include
       catalog with order
   </p>

   <p>Ship to: <br />
      Name: <input type=“text” name=“name” /> <br />
      Address: <input type=“text” name=“address” /> <br />
      City: <input type=“text” name=“city” /> <br />
      State: <input type=“text” name=“state” /> <br />
      Zip: <input type=“text” name=“zip” /> <br />

   <p>Customer Comments: <br />
      <textarea name=“comments” rows=“10” cols=“48”></textarea>
   </p>

   <p>
      <input type=“submit” value=“Order Now” />
   </p>

</form>
```

此处，我们为每一个选项都添加了一个复选框。注意这些复选框共享一个名字“extras[]”……

……不过复选框的值不同。

我们使用cheked 属性定义默认的选项。你也可以在多个复选框中添加cheked 属性。

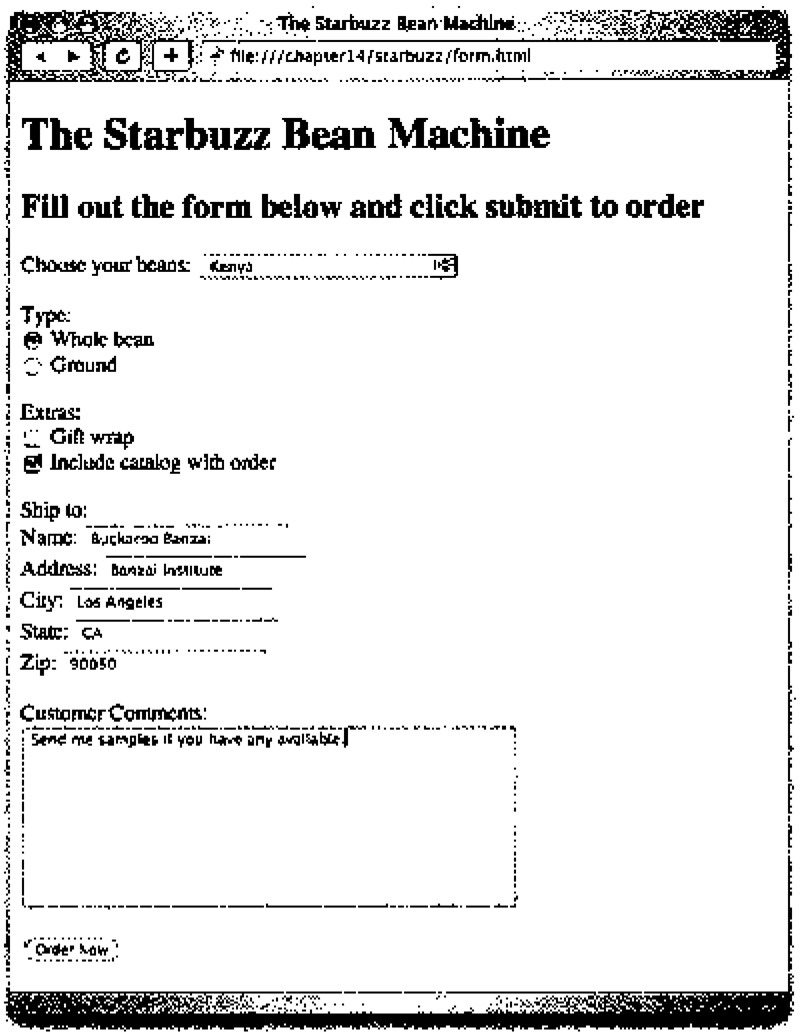
和单选框按钮一样，在复选框的右边对它们进行标记。

文本区在这儿。

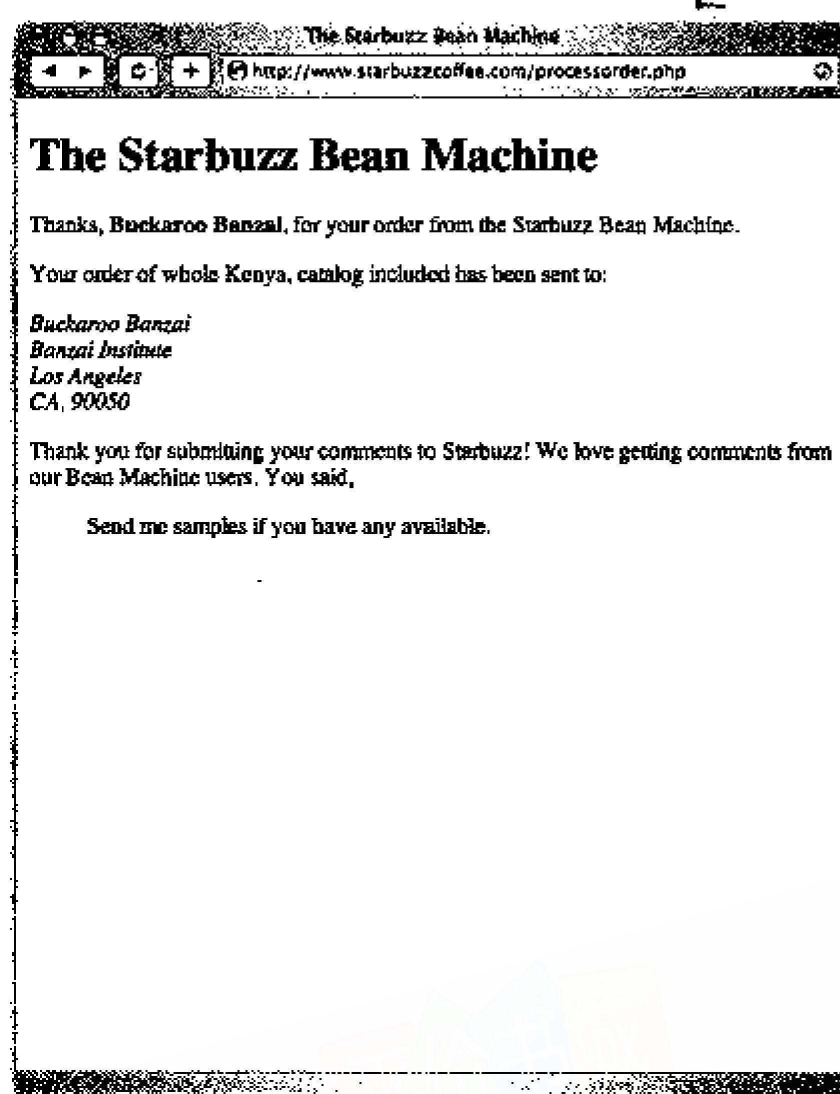
我们想定义一个高度为10个字符，宽度为48个字符的文本区。

最终运行测试

保存修改，再一次加载页面，看看新的表单，难道你认为它看起来不美观吗？



这就是提交表单后收到的反馈。Web应用程序接收了来自网页的所有表单数据，然后将这些数据一起反馈给网页。看看你能否找到你提交的所有表单数据。



确保尝试各种发送表单的组合（选了或没选礼物包装的，选了或没选商品目录的，不同咖啡豆的，等等），然后看看显示的结果。



不论你相信与否，“extras[]”对于表单元素来说，是一个非常正当的名字。

尽管它是非常正当的，但它看上去和正常的不一样，不是吗？处理办法：透视XHTML,这是一个普通的元素名。当名字中有方括号时，它不会对浏览器产生任何的影响。

那么我们为什么使用有方括号的名字呢？是因为，用脚本语言书写的处理顺序.php Web应用程序如同一个小暗示，它暗示了一个表单变量可以拥有多个值。提供这种暗示的方法就是在名字的末端添加一个方括号。

所以，通过对XHML的透视，你大可以忽略这些细节。不过你可能要将这个知识点塞进你的大脑，万一将来你要用PHP Web应用程序编写表格呢。



扮演浏览器

下边是一个XHTML菜单,右边是用户输入表单中的数据。你的任务就是扮演浏览器。你需要将用户输入的值与表单中各个元素相匹配。做完这个练习以后,看一下这一章结尾部分的答案,检查你所匹配的表单元素名与值是否正确。

```
<form action="http://www.chooseyourmini.com/choice.php" method="POST">
   <p>Your information: <br />

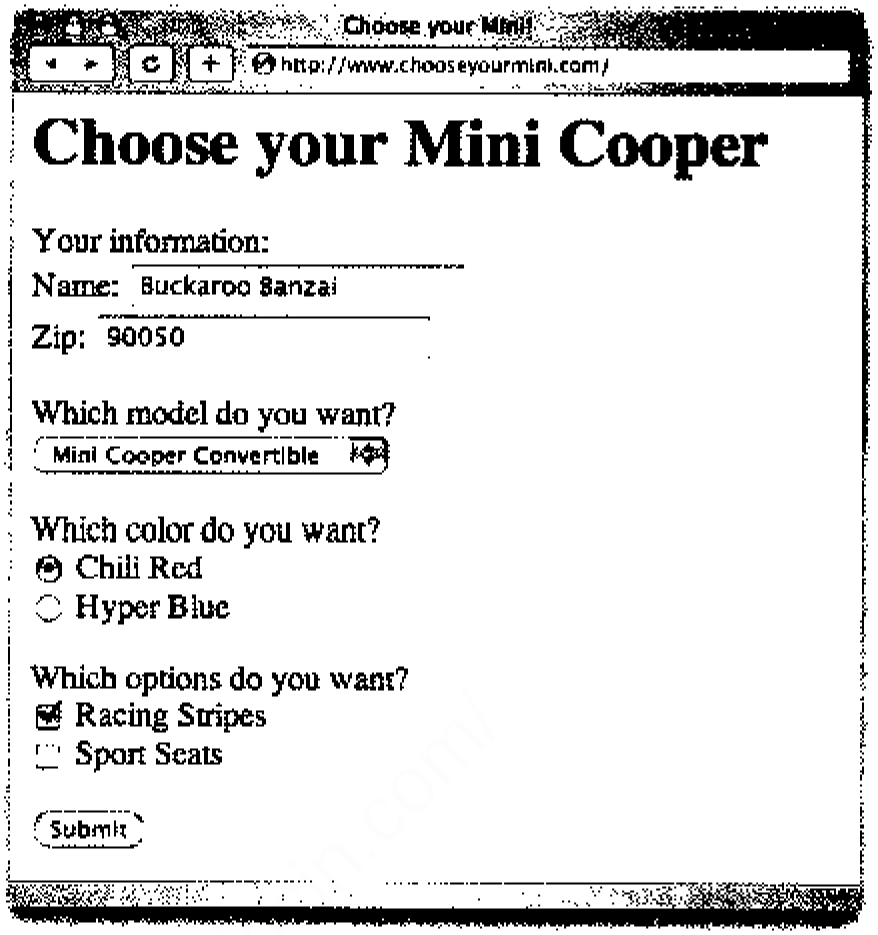
      Name: <input type="text" name="name" /><br />
      Zip: <input type="text" name="zip" /><br />

   </p>
   <p>Which model do you want? <br />
     <select name="model">
         <option value="cooper">Mini Cooper</option>
         <option value="cooperS">Mini Cooper S</option>
         <option value="convertible">Mini Cooper Convertible</option>
     </select>
   </p>
   <p>Which color do you want? <br />
     <input type="radio" name="color" value="chilired" /> Chili Red  <br />
     <input type="radio" name="color" value="hyperblue" /> Hyper Blue
   </p>
   <p>Which options do you want?  <br />
     <input type="checkbox" name="caroptions[]" value="stripes" /> Racing Stripes
     <br />
     <input type="checkbox" name="caroptions[]" value="sportseats" /> Sport Seats
   </p>

   <p>
     <input type="submit" value="Order Now" />
   </p>

</form>
```

表单在这儿。



这个表单已经填满了。

为每个表单名匹配一块表单数据，并把你的答案写在这上面。

name = "Buckaroo Banzai"

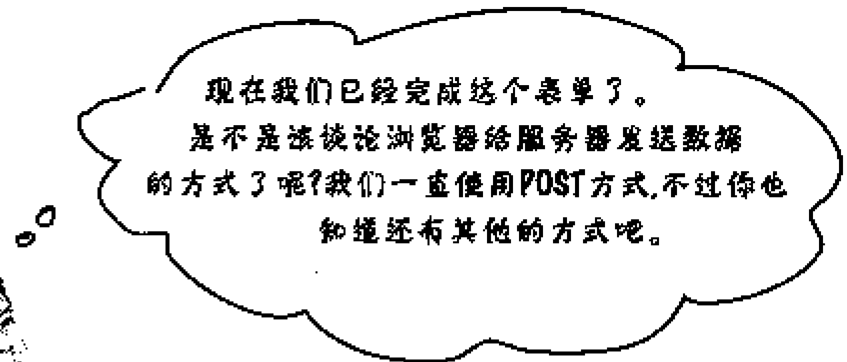
zip = ________

model = ________

color = ________

caroptions[] = ________

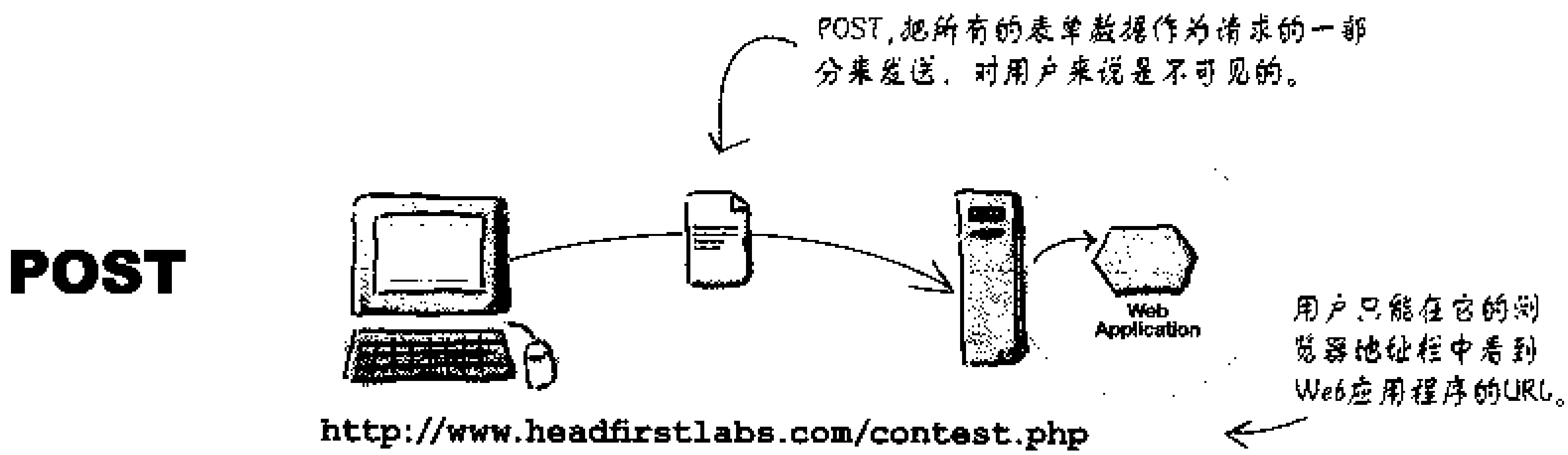
附加信用卡……

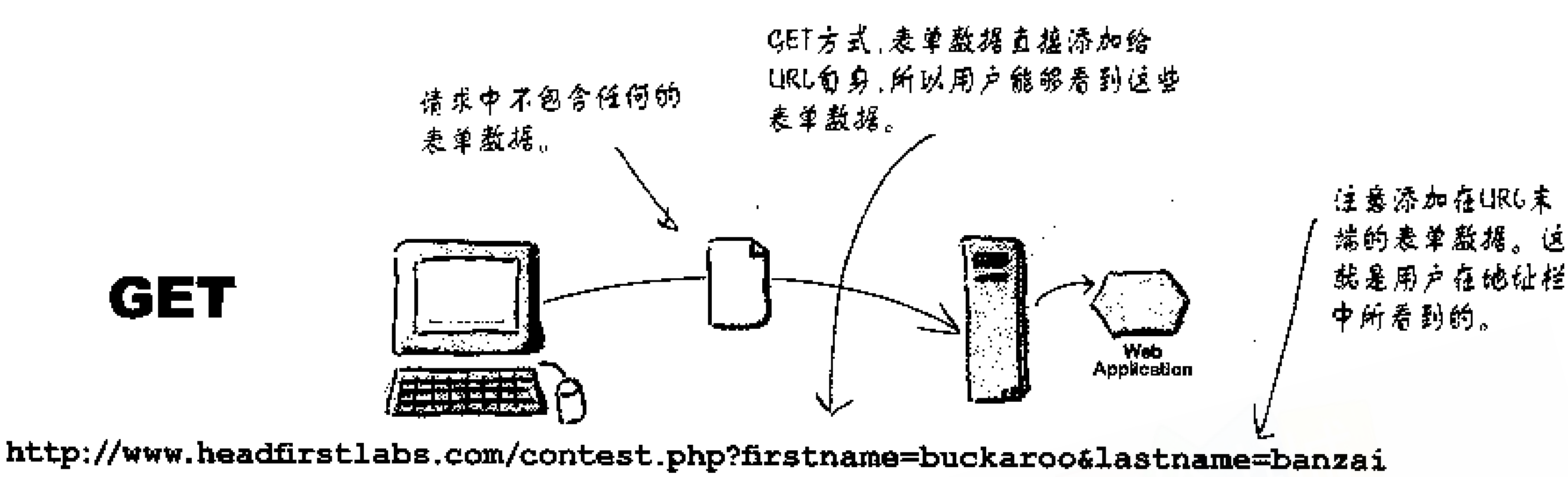




浏览器使用的两种基本方式：POST和GET。

POST和GET都能完成相同的事情——把表单数据发送给服务器——不同的是它们使用的方式。POST，把表单变量打包后隐藏在后台发送给服务器；GET也把表单变量打包,不过在它向浏览器发送请求之前，附加在URL的末端部分。





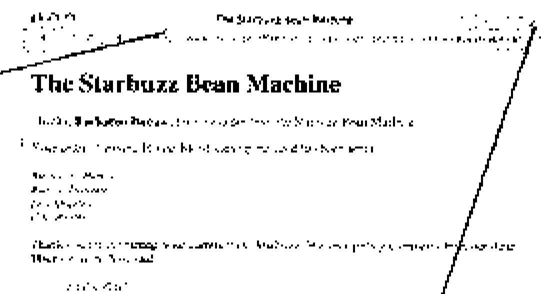
运行观察GET

没有比动手运行更好的方法来了解GET了。打开“form.html”文件,做以下改动：

只把“POST”修改为“GET”。

```
<form action="http://www.starbuzzcoffee.com/processorder.php" method="GET">
```

保存并加载网页，输入信息后提交。你将会看到像这样的页面：



在浏览器中看到的URL。

```
http://www.starbuzzcoffee.com/processorder.php?beans=Kenya&beantype=gro
und&extras%5B%5D=catalog&name=Buckaroo+Banzai&address=Banzai+Instit
ute&city=Los+Angeles&state=CA&zip=90050&comments=Great+coffee
```

你可以看到每一个表单元素名及表单元素的值，就在URL中。

注意，浏览器将表单数据翻译成各式各样的字符，比如间隔符。Web应用程序在接收到表单数据后会自动解码。

there are no Dumb Questions

问： 如果我向服务器发送东西，为什么要叫做“GET”呢？

答： 好问题。浏览器的主要任务是什么呢？是从服务器获取网页。那么，当你使用GET时，浏览器像往常一样使用一般方式获取一个网页，除此之外，在表单事务中，它把较多数据附加在URL的末端。另外，浏览器还扮演了一个普通的请求角色。

而对于POST，换句话说，浏览器实际上是创建了一个小型的数据包，然后发送给服务器。

问： 那么，为什么我要用POST而不是GET,或者 vice versa?

答： 这是两个大的区别，也是真正的问题所在。如果你希望用户能够标记网页，也就是提交表单后的结果，那么你就使用GET，因为如果网页用POST方式的结果返回，就不能标记网页了。什么时候需要使用GET方式呢？当你的Web应用程序返回一个搜索结果列表时，你可能希望用户能够标记返回的结果，这样用户不用再填满表单就能看到结果了。

另一方面，如果你是处理订单的Web应用程序，那么你不希望你的用户标记这个网页（否则，用户每次返回所标记的页面，订单会被再次提交）。

有一种情况，你“绝不”希望使用GET方式：当表单中的数据是私人数据时，比如信用卡或者密码。由于URL以清晰的方式显示，那么通过浏览你的浏览器历史或者GET方式以某种途径获取标记网页，他人可以很容易地获取你的私人信息。

最后，如果你使用的是<textarea>，你应该用POST，因为你可能会发送大量的数据。GET请求方式限制只能使用256个字符，而POST在发送的数据页面大小上没有限制。

Sharpen your pencil

GET 还是 POST

对每一个描述，在合适的方式上画圈。如果你觉得两者皆可，可以两个都画圈。准备捍卫你的答案吧……

GET **POST** 输入用户名和密码的表单。

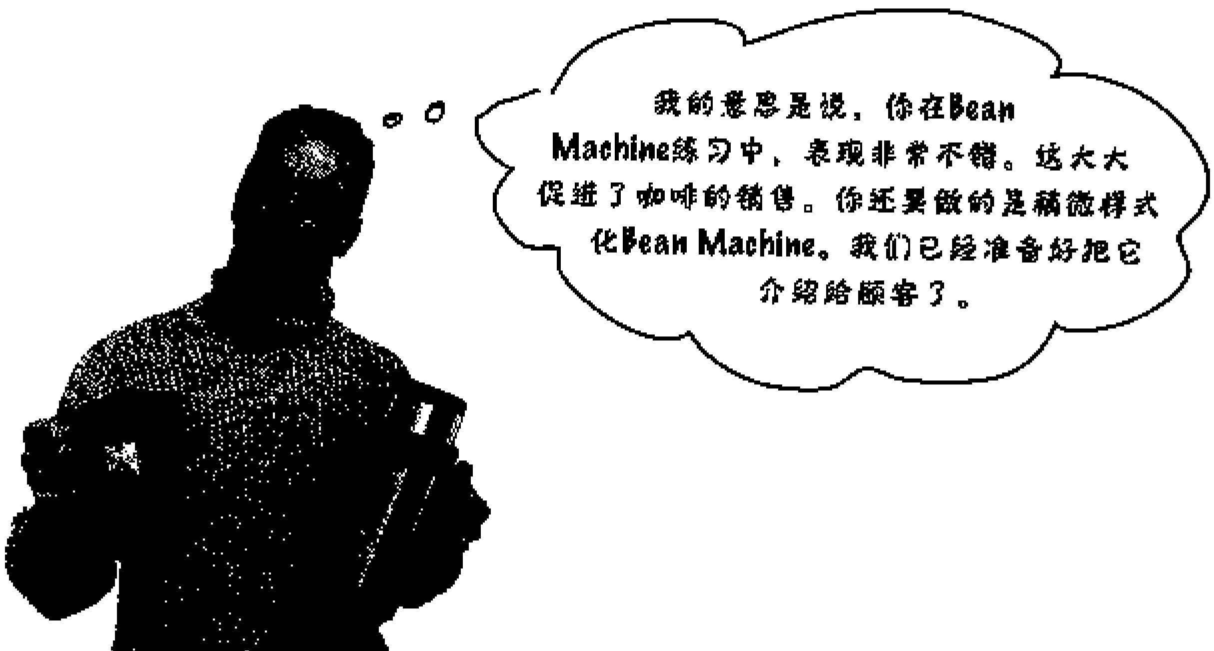
GET **POST** 预定CD的表单。

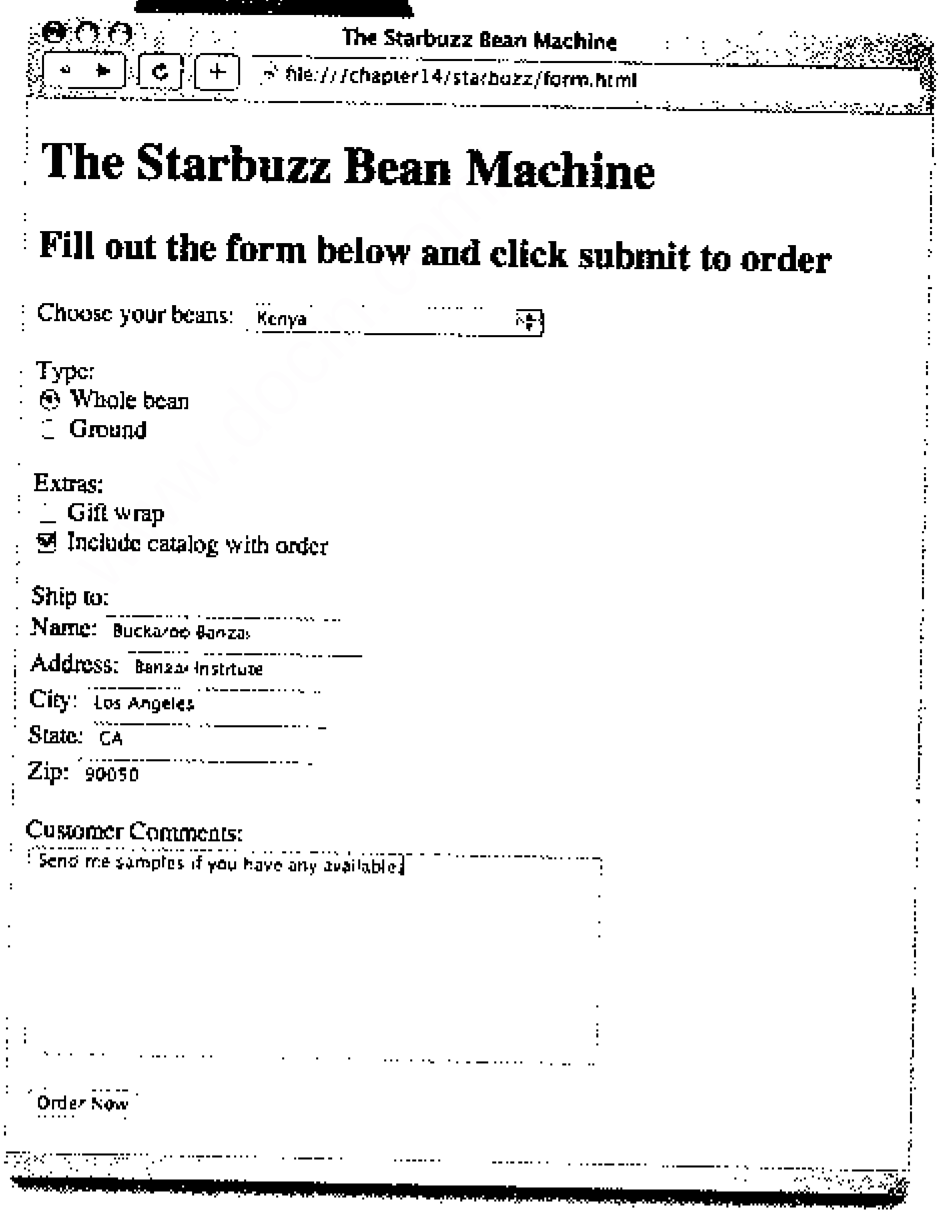
GET **POST** 查询时事的表单。

GET **POST** 邮寄复习书的表单。

GET **POST** 通过身份证号码获取收益的表单。

GET **POST** 发送用户反馈的表单。

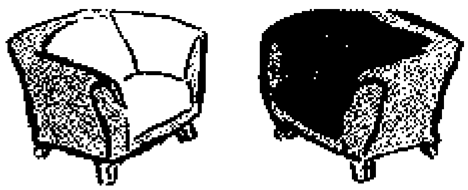




BRAIN POWER

已经学过了XHTML和CSS，怎样运用这些方法样式化这个表单？

围炉夜话



今夜话题：表格和CSS争论如何布局表单。

表格：嘿，CSS，发生什么事情了？

CSS：这一章中你究竟干了些什么，表格？

表格：你什么意思？我顺便过来帮一下那些表单，使它们做得更好一点。它们看上有一点儿……粗糙,如果你问我的话我会这么说。

CSS：我赞成表单应该在外观方面做一些修补，不过那是我的工作。你负责的是表格式数据，记得吗？

表格：是的，一些人把表单元素当做表格式数据，你也知道的。除此之外，在表单外观方面，我比你做得更好。

CSS：你说谁做得好？

表格：我啊。通过把所有的标签和表单元素适当地对齐，我可以让表单更加地美观和紧凑。

CSS：你知道，我也可以放置这些元素。读者在相应的地方已经读过这些内容了，他们知道怎样用CSS使元素“全都适当地对齐”的方法。不管怎样，尽管你可以适当地对齐表单元素，你一定不能做比如：添加颜色、补白，改变字体家族这类事情。

表格：是的，我承认我不能够添加那些小的特别的你能够做的事情，不过那些事情根本就不重要。那好比蛋糕上的糖一样。制作贴近用户的表单的关键是使得标签和元素能够有效地显示出来，所以不存在是用什么修饰的问题。用户事实上并不关心这类事情。

CSS： 你根本就不知道你在说什么。表单应该和网站其余部分在视觉上、感觉上相匹配。用户对于他们是否需要填满表单产生疑惑，因为表单看上去不像网站的一部分。

表格： 我知道。不过当遇到要适当地布局表格时，那就是我的工作了。上次我看到有人试图用CSS布局表格时，程序中处处布满了<div>和<span>，简直太乱了！光是看看我都头痛。

CSS： 那么，你的XHTML中那些乱扔的<tr>和<td>呢？那没有什么不同。

表格： 至少我的表格行和数据单元格很容易分清楚在哪里结束，而对于你玩的位置把戏，我不知道元素是在哪里结束的。

CSS： 很明显，你没有看过第12章。我不接受那些表格式数据的表单元素……它们是XHTML元素，而非数据。

表格： 不过，用户却把数据输入表单控件中，不是吗？表单用来从使用表单的用户那里收集数据。怎么不是表格式数据呢？

CSS： 嗯……我想我可以理解……不过用表格来做布局看上去是如此的荒谬。布局就是外观，外观就是我的工作。

表格： 我说，如果我比你做得好，那么，就使用我吧。我很荣幸能做这件事。

CSS： 比我干得好？你还差得远呢……

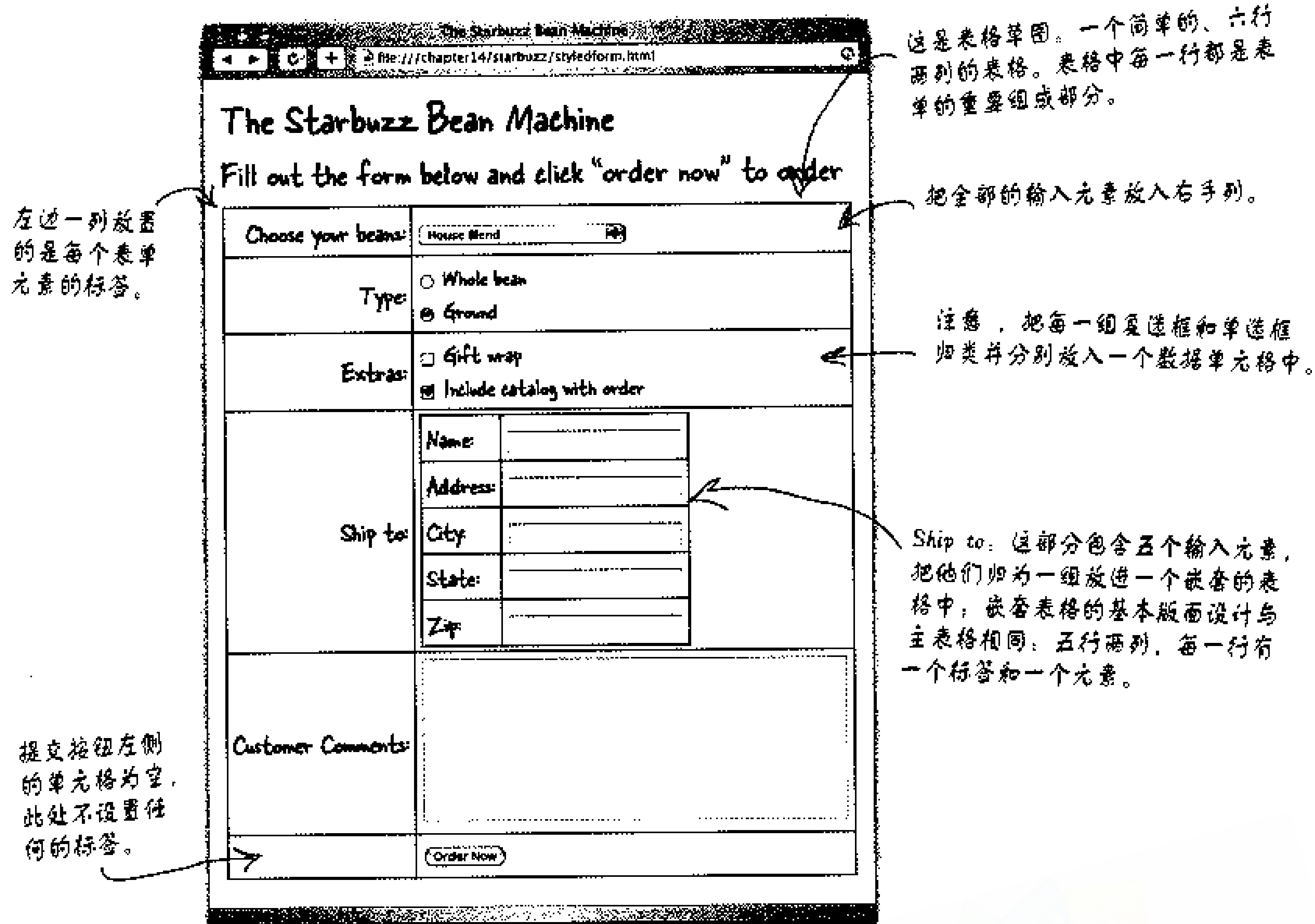
表格： 听我说，我们为什么不让读者决定呢？

用不用表格？这是个问题……

你将发现用户站在问题的两个角度上。你是使用CSS布局表单呢？还是表格？残酷的事实是使用CSS布局表单是非常困难的。如果你愿意折损空间、花费时间来用CSS布局表单的话，我们也会很高兴地为你让路，用钦佩的眼光旁观的。然而，大多数表单在布局上都是表格式的，所以为什么要用CSS格式，而不采用表格布局表单呢？那样的话，不论是表单，还是表格，我们都可以做得很好。

开始布局吧……

我们先把表单放入表格中。看一看下面的草图，表格中的表单搭配得很合适甚至很好，这看起来像个表单了而不是输入元素的杂碎集合。还有，注意我们在“Ship to:”部分使用了一个嵌套的表格。



把表单元素放入表格



现在，你已懂得如何把表单元素组织在一个表格中，你需要将你的XHTML书写技巧运用到你的实际测试中。那么开始写吧。

只是开个玩笑。我们不是让你书写整个的表格……毕竟这一章主要讲的还是表单,而不是表格。表格部分我们已经为你写好了，就在“chapter14/starbuzz/”文件夹下的“styledform.html”文件中。尽管这个文件看上去很复杂，不过也不是那么糟糕。我们已经在文件中添加了一些注释，用来指出其中的主要部分。

这是表单元素，我们不需将此放入表格中。

```
<form action="http://www.starbuzzcoffee.com/processorder.php" method="POST">
```

这是表格的开始。

主表格的每一行都包括了两个数据单元格：其中一个<th>设置标签，而<td>设置表单元素。

```
<table>
   <tr>
      <th>Choose your beans:</th>
      <td>
         <select name="beans">
            <option value="House Blend">House Blend</option>
            <option value="Bolivia">Shade Grown Bolivia Supremo</option>
            <option value="Guatemala">Organic Guatemala</option>
            <option value="Kenya">Kenya</option>
         </select>
      </td>
   </tr>
```

表单的每一部分单独为表格的一行。

```
   <tr>
      <th>Type:</th>
      <td>
         <input type="radio" name="beantype" value="whole" />
         Whole bean
         <br />
         <input type="radio" name="beantype" value="ground" checked="checked" />
         Ground
      </td>
   </tr>

   <tr>
      <th>Extras:</th>
      <td>
         <input type="checkbox" name="extras[]" value="giftwrap" />
         Gift wrap
         <br />
         <input type="checkbox" name="extras[]" value="catalog" checked="checked" />
         Include catalog with order
      </td>
   </tr>
```

对于咖啡豆选择菜单，“beantype”为单选框按钮，“extra”为复选框，所有这些菜单表单元素都分别放在一个数据单元格中。

至于shiping数据，我们则创建了一个嵌套的表格——一个单元格中放入一个表格。这样便可以对齐表单中"ship to"部分的文本<input>标签。

```
<tr>
   <th>Ship to:</th>
   <td>
      <table>
         <tr>
            <td>Name:</td>
            <td>
               <input type="text" name="name" value="" />
            </td>
         </tr>
         <tr>
            <td>Address:</td>
            <td>
               <input type="text" name="address" value="" />
            </td>
         </tr>
         <tr>
            <td>City:</td>
            <td>
               <input type="text" name="city" value="" />
            </td>
         </tr>
         <tr>
            <td>State:</td>
            <td>
               <input type="text" name="state" value="" />
            </td>
         </tr>
         <tr>
            <td>Zip:</td>
            <td>
               <input type="text" name="zip" value="" />
            </td>
         </tr>
      </table>
```

这是shiping数据嵌套表格的结束部分。

```
   </td>
</tr>

<tr>
   <th>Customer Comments:</th>
   <td>
      <textarea name="comments" rows="10" cols="48"></textarea>
   </td>
</tr>

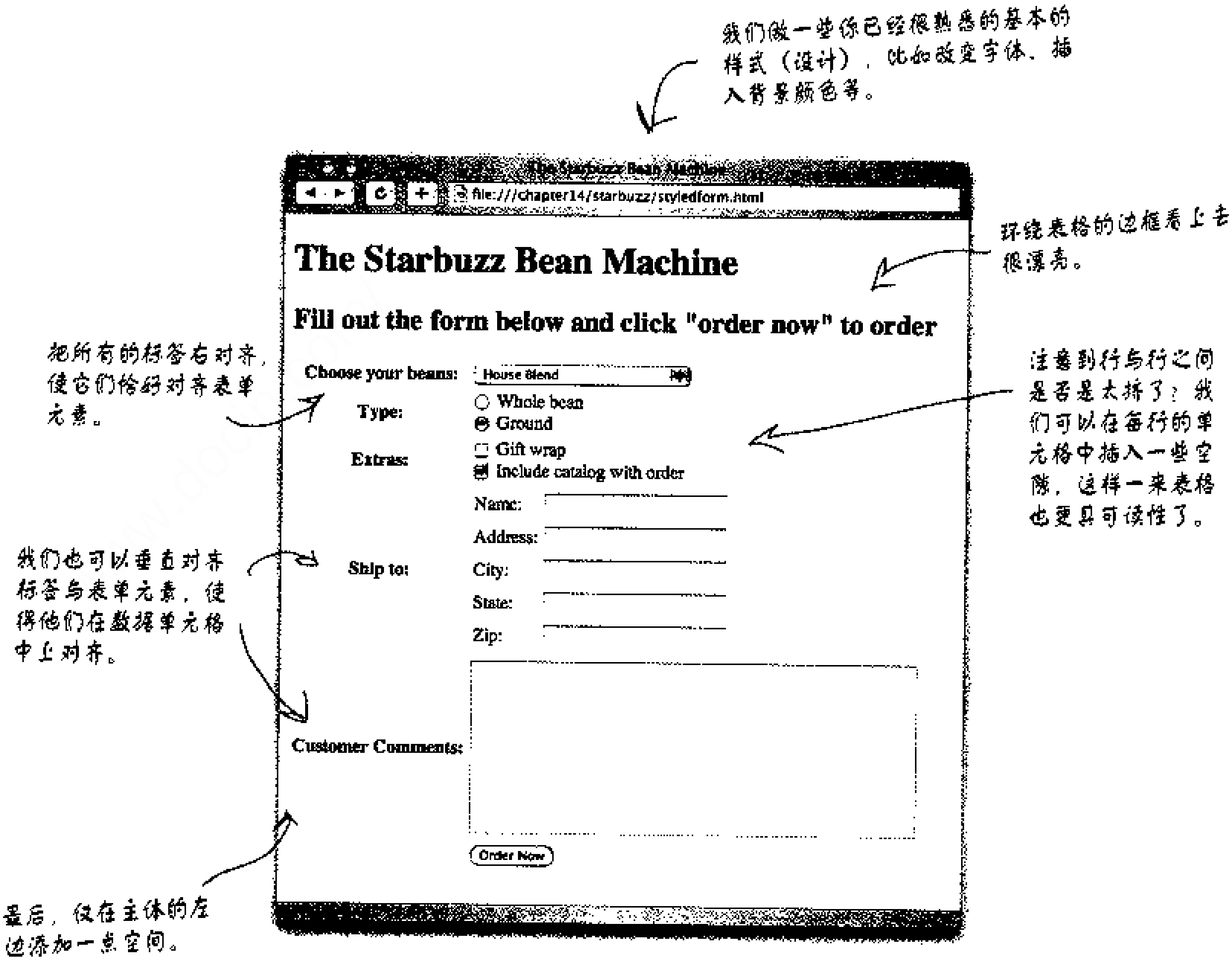
<tr>
   <th></th>
   <td><input type="submit" value="Order Now" /></td>
</tr>
</table>
</form>
```

这是主表格中的<textarea>文本行及<input>提交行。

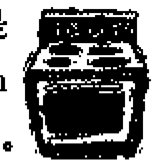
测试表格式表单

在浏览器中打开“styledform.html”文件，然后看一看以表格格式显示的Starbuzz Bean Marchine表单。表单更好看了，不是吗？所有的标签、表单元素都对齐了，看上去更加专业化了。

现在，我们要用CSS在各处修改一番使它变得更加完美一些。就让我们看一看哪些地方需要你修改。



用CSS样式化表单和表格



待烘烤CSS

我们只需在XHTML上添加一些样式规则，而且我们也将这么做。由于这个表单是Starbuzz网站的一部分，我们将重新使用“starbuzz.css”样式表中的样式。在Bean Marchine表单中添加新的样式规则便可创建一个新的样式表“styledform.css”。CSS中这些规则对你来说都很熟悉了。我们不会对表格或者表单使用任何特殊的规则,都是一些你在上一章中使用过的规则。

你可以在“Chapter14/starbuzz”文件夹下的“styledform.css”中找到这个CSS。

在Starbuzz CSS样式基础上，把主体字体设置为san-serif 字体，添加Starbuzz背景图像，给主体添加边界。

```
body {
    background: #efe5d0 url(images/background.gif) top left;
    font-family: Verdana, Helvetica, Arial, sans-serif;
    margin: 20px;
}
```

所有页面中的元素都继承了这些字体属性，包括表格文本、表单文本。

```
table {
    border: thin dotted #7e7e7e;
    padding: 10px;
}
```

添加环绕表格的边框，添加表格内容与边框之间的补白。

```
th {
    text-align: right;
    vertical-align: top;
    padding-right: 10px;
    padding-top: 2px;
}
```

这个表单标签位于表格表头中。我们想要向上向右对齐表单标签，这样的话就可以很工整地对齐右列的表单元素。我们还在表格表头中插入一些补白，在标签的上边和右边分别插入一些空隙。

```
td {
    vertical-align: top;
    padding-bottom: 15px;
}
```

数据单元格的内容默认向左对齐，那就是我们所要的，同时我们还需要垂直对齐来匹配表格表头。我们还在这添加了一些补白以增加行与行之间的空隙。

```
table table {
    border: none;
    padding: 0px;
}

table table td {
    text-align: right;
    padding-bottom: 0px;
}
```

这两段程序重载了上面设置的table和td规则属性。为什么呢？因为，我们希望嵌套的表格不含有边框，空间要更加紧凑，所以撤消了补白。不过我们仍想右对齐嵌套数据单元格中的表单标签（这些表单标签不在表格表头中，所以它们不会使用上面的th规则来对齐）。

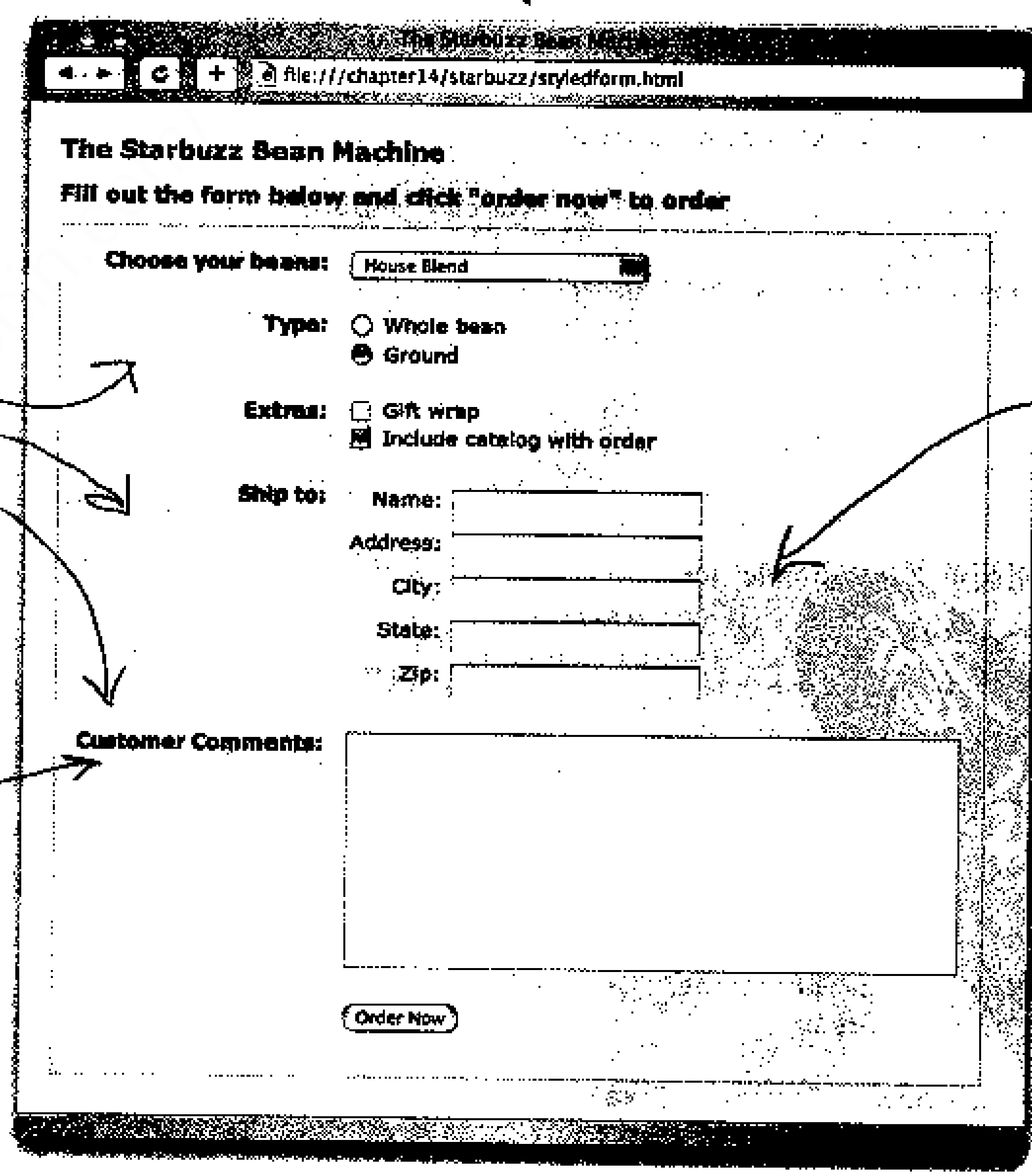
最终测试

在“styledform.html”文件的XHTML**<head>**中添加两个**<link>**元素,链接来自第12章的“Starbuzz.css”样式表和一个新的样式表“styledform.css”。确定你已经使规则正确无误了：首先链接“Starbuzz.css”文件,其次链接“styledform.css”。链接完两个样式表后,保存并加载网页。你会在浏览器中看到一个时髦的,多样式版本的Starbuzz Bean Marchine。

如果你想提高一点儿你书写XHTML和CSS的技术,看看你是否能够把Starbuzz标题和页脚添加到Bean Marchine网页中,那看上去真的很漂亮。

哇,小小的样式出现了惊人的效果!

现在的Bean Marchine表单能够与Starbuzz网站的其他部分相匹配了。



标签与表单元素的顶端对齐,同时也右对齐。这种对齐方式更容易看清楚哪个标签应该对应哪个控件。

行与行之间的空隙起了很大作用,而且使得表单的可读性更好。

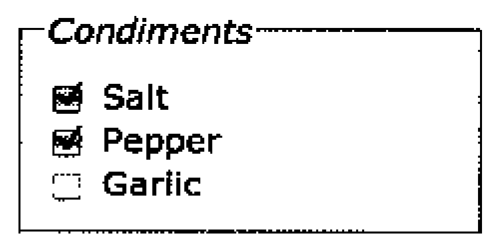
注意,嵌套的表格没有边框,而且单元格之间的空隙很小,那是因为嵌套的表格规则重载了主表格的属性(规则)。

表单中还有哪些元素？

我们已经介绍了你在表单中经常使用的全部元素，不过还有一些元素你可能认为有必要添加到你的表单中。所以我们在这里介绍这些以备将来你制作自己的表单时会用到。

Fieldsets 和legends

当表单规模开始变大时，把元素组合起来会有帮助。你可能会使用**<div>**和CSS完成组合，不过XHTML也提供了一个**<fieldset>**元素用来组合普通元素。**<fieldset>**中另一个有用的元素是**<legend>**。看一看它们是如何搭配工作的：



这就是浏览器中所看到的fieldset和legend。你会发现在不同的浏览器上，显示效果也不同。

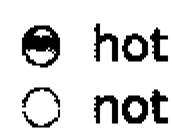
<fieldset>元素包围了一组输入元素。

<legend>为每个组合提供一个标签。

```
<fieldset>
   <legend>Condiments</legend>
      <input type="checkbox" name="spice" value="salt" />
         Salt  <br />
      <input type="checkbox" name="spice" value="pepper" />
         Pepper <br />
      <input type="checkbox" name="spice" value="garlic" />
         Garlic
</fieldset>
```

标签

到现在为止，你已经用简单文本（simple text）标注了表单，不过，XHTML也提供了一个**<label>**元素。这个元素提供了一长串的网页结构信息，允许你更容易地使用CSS来样式你的标签，还可以帮助视力受损者用屏幕读取器修正信息。



默认的标签看上去不同于纯文本。不过，它们在可读性上有很大的区别。

要使用<label>元素，首先应该在表单元素中加入id属性。

```
<input type="radio" name="hotornot" value="hot" id="hot" />
<label for="hot">hot</label>

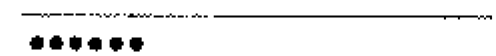
<input type="radio" name="hotornot" value="not" id="not" />
<label for="not">not</label>
```

添加一个<lable>并设置一个与id相一致的for属性。

你可以对任何的表单元素使用<label>元素。

口令

除了输入的文本被隐藏之外，**<input>**口令元素与**<input>**文本元素功能相同。隐藏输入文本对于那些要求你输入口令、秘密代码、或者你不希望他人看到的敏感信息的表单是很有用处的。记住，从浏览器发送至Web应用程序的表单数据是不安全的，除非你做了安全措施。想了解更多的安全问题，联系你的服务器代理商。



除了输入的文本被隐藏外，<input>口令元素与<input>文本元素非常地相似。

```
<input type="password" name="secret" />
```

文件输入

这里有一个全新的、我们从未谈及的input元素。如果你需要将整个文件发送给Web应用程序，你将再次使用**<input>**元素，只不过把它的type设置为“file”。设置完毕后，**<input>**元素会创建一个控件，控件允许选择文件，而且当表单被提交,文件内容连同余下的表单元素一起发送给服务器。记住，你的Web应用程序希望接收一个上传的文件，还要注意，你必须使用POST方式使用此元素。



这是文件输入元素在两个不同的浏览器中的显示的效果。

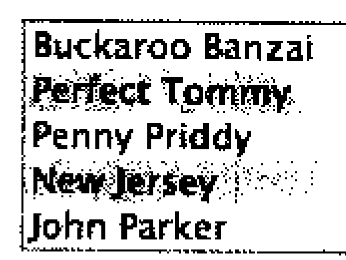
```
<input type="file" name="doc" />
```

只需将元素的type设置为file，就可创建一个文件<input>元素。

多选属性

这不是一个新元素，却是你所知的一个元素的新用法。在**<select>**元素中添加一个值为“multiple”的**multiple**属性，就可把单选菜单变为多选菜单。你可获取一个可替代下拉菜单的多选菜单，这个菜单在屏幕上显示出所有的选项。（如果选项太多的话，就会出现一个滚动条）；你可以选用按下Ctrl(window)或者Command(Mac)来选择多个选项。

通过multiple属性，你可以一次选择多个选项。



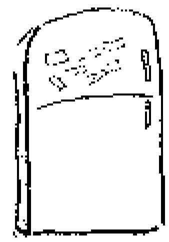
只需添加一个值为“multiple”的multiple属性，就可将单选菜单转化为多选菜单。

```
<select name="characters" multiple="multiple">
    <option value="Buckaroo">Buckaroo Banzai</option>
    <option value="Tommy">Perfect Tommy</option>
    <option value="Penny Priddy">Penny</option>
    <option value="New Jersey">Jersey</option>
    <option value="John Parker">John</option>
</select>
```

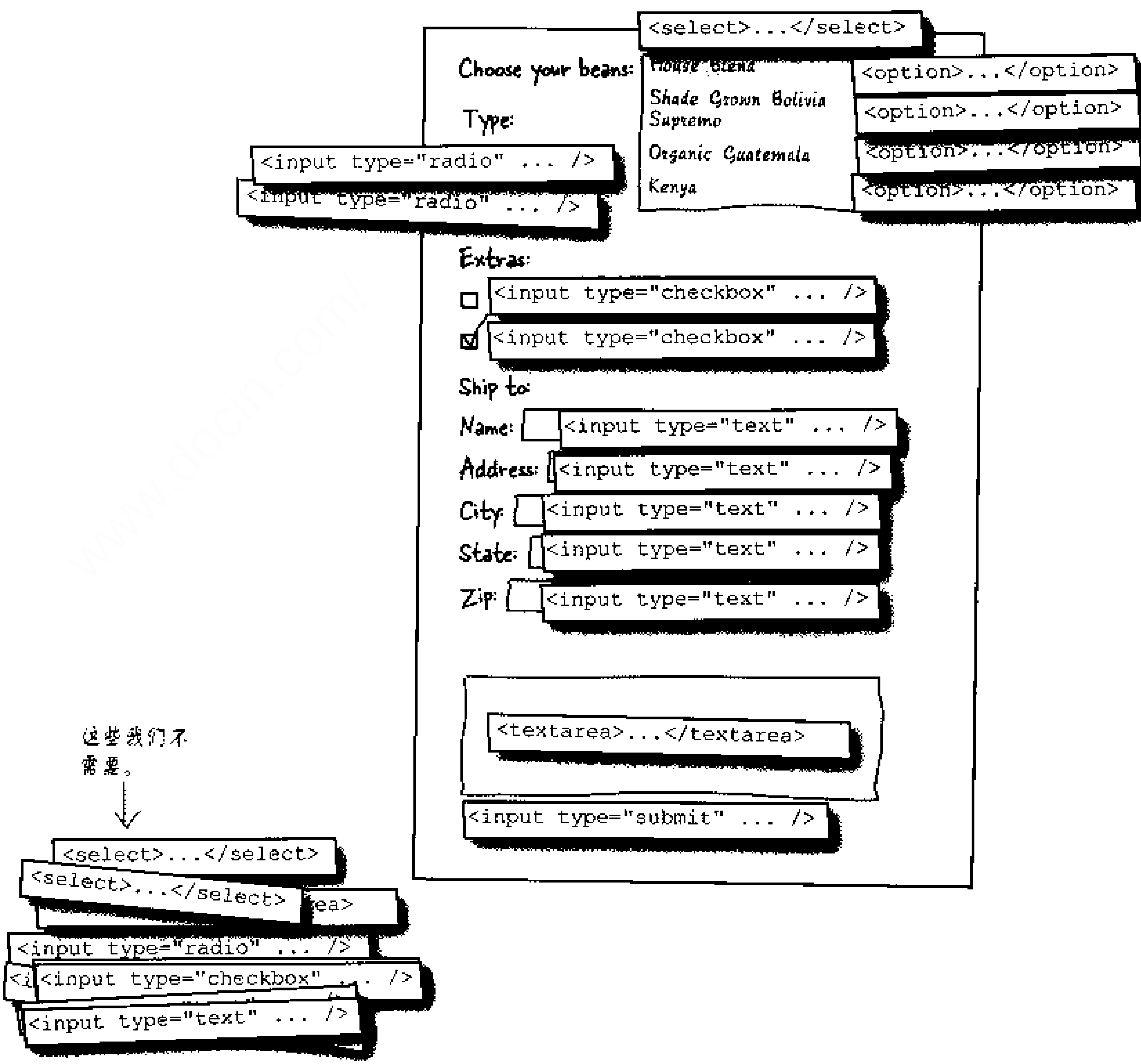
要点

- <form>元素定义了一个表单，所有的input元素都放在<form>元素中。
- Action属性包含了Web应用程序的URL。
- Method属性包括两种发送表单数据的方法：一种是POST，一种是GET。
- POST将数据打包并将数据包作为请求的一部分发送。
- GET也将数据打包并将数据包附在URL上。
- 当表单数据为私有或者数量大时，使用POST方式，比如<textarea>或者文件<input>元素。
- 对于可能要标注的请求，使用GET。
- <input>元素能够演变为网页中许多不同的input控件，主要取决于type的属性值。
- Type为text,创建一个单行文本输入框。
- Type为submit,创建一个提交按钮。
- Type为radio,创建一个单选框按钮。所有单选框按钮名字相同，这些单选框组成一个互助互斥的按钮组。
- Type为checkbox，创建一个复选框控件。多个复选框共享一个复选框名字，并可得到一组选项。
- <textare>元素创建一个多行文本输入区。
- <select>元素创建一个包括一个或多个<option>元素的菜单。<option>元素定义了菜单的项目。
- 如果把文本放入<textarea>元素的内容中，那么在网页上将会当成文本区控件的默认文本。
- Text<input>元素中的value属性，可以单独为单行输入文本框设置value值。
- 在提交按钮上设置value属性，可改变按钮上的文字。
- 当网页表单被提交后，表单数据值与相应的表单数据名匹配成对，然后把所有的表单数据名与值发送给服务器。
- 表格常用作表单的版面设计，给表单一个表格式的结构。布局后，CSS为表单的外观进行样式化，比如颜色、字体等。
- XHTML允许使用<fieldset>元素组合表单元素。
- <lable>元素以一种很简单的方式给表单元素附加标签。

标记帖答案



你的任务是提取表单元素，并把它们放到图中相应的控件上方。你不需要使用下面所有标记帖就可以完成这个任务。左边已经给出一些表单元素。下面是参考答案。





name = "Buckaroo Banzai"

zip = "90050"

model = "convertible"

color = "chilired"

caroptions[] = "stripes"

练习答案

Sharpen your pencil 答案

GET 还是 POST

关于每一个描述，在更合适的方式上画圈。如果你觉得两者皆可，可以两个都画圈。准备捍卫你的答案吧……

- GET **(POST)** 输入用户名和密码的表单。
- GET **(POST)** 预定CD的表单。
- **(GET)** POST 查询时事的表单。
- GET **(POST)** 邮寄复习书的表单。
- GET **(POST)** 通过身份证号码获取收益的表单。
- GET **(POST)** 发送用户反馈的表单。

练习答案



如果你将一个值为“checked”的checked属性添加到单选框input元素中，那么浏览器在显示表单时，那个选项就会被默认为已选状态。在ground 的单选框<input>元素中添加checked属性，并测试这个网页。下面是参考答案。

这就是文件“form.html”表单部分。

```
<form action="http://www.starbuzzcoffee.com/processorder.php" method="POST">
  <p> Choose your beans:

      <select name="beans">

        <option value="House Blend">House Blend</option>

        <option value="Bolivia">Shade Grown Bolivia Supremo</option>

        <option value="Guatemala">Organic Guatemala</option>

        <option value="Kenya">Kenya</option>

      </select>

  </p>

  <p>Type: <br />

      <input type="radio" name="beantype" value="whole"  /> Whole bean <br />
      <input type="radio" name="beantype" value="ground" checked="checked" /> Ground

  </p>
  <p>Ship to: <br />
      Name: <input type="text" name="name" /> <br />
      Address: <input type="text" name="address" /> <br />
      City: <input type="text" name="city" /> <br />
      State: <input type="text" name="state" /> <br />
      Zip: <input type="text" name="zip" /> <br />
      <input type="submit" value="Order Now" />
  </p>
</form>
```

这个新的属性选择“ground”单选框按钮。



恭喜！
已经学到了这本书的最后部分了。

当然，这里还有剩余部分。

还有索引。

还有版本记录。

还有网站……

真的没有漏掉什么。

附录A：剩余部分

排名前十的主题（本书还未涉及）



我们介绍了许多基础知识，而且你也差不多读完了此书。

我们会想念你的，不过在让你走之前，如果就这样在你毫无准备的情况下将你推向社会，我们觉得不太合适。我们不可能把所有你需要了解的知识都包含在这一小节中。事实上，我们最初是这么做的，把字体尺寸减少到.00004，这样的话这一小节就包括了所有你需要知道的关于XHTML和CSS的知识（其他章节中没有讲到的）。这样确实能够把所有内容装进去，不过恐怕也没有人会读了。所以，我们扔掉了大部分内容，只保留了最好的排名前十的主题。

这的的确确是这本书的末尾了，当然除了索引部分（必读）。

#1 更多选择符

你已经学过了许多常用选择符，这里有一些可能你也想要了解的选择符……

伪元素选择符

你对伪类已经有一个全面的了解了。其实伪元素选择符与伪类选择符很相似。伪元素选择符可以用来选择元素的一部分，这些元素要是用**<div>**和**<span>**元素或其他的方式来处理不太方便。比如说**:first-letter**伪元素选择符可以用来选择一个块元素文本的第一个字，允许你设置首字大写和首字下沉。还有一个叫做**first-line**的伪元素选择符，用来选择某一段的第一行文字。下面，我们就是用这两个伪元素选择符来选择**<p>**元素的首字和首行：

```
p:first-letter {
        font-size: 3em;
}

p:first-line {
        font-style: italic;
}
```

伪元素选择符使用的语法与伪类相同。

我们在这里设置了一个大号的段落首字和一行段落斜体字。

属性选择符

现在的浏览器不能很好地支持属性选择符，不过在不久的将来就能广泛地支持了。属性选择符，顾名思义，你能基于属性值选择元素。你可以这样使用它们：

此选择符选择XHTML中包含width属性的所有图像。

```
img[width] { border: black thin solid; }

img[height="300"] { border: red thin solid; }

image[alt~="flowers"] { border: #ccc thin solid; }
```

此选择符选择包含heigh属性值为300的所有图像。

此选择符选择含有alt属性并且值为"flowers"的所有图像。

基于兄弟进行选择

你也可以基于前面的兄弟选择元素。例如：你想要选择前面的有一个**<h1>**元素的段落，那么你可以用这个选择符。

首先书写前面的元素，接着是‘+’符号，最后是兄弟元素。

```
h1+p {
  font-style: italic;
}
```

此选择符选择紧跟着一个<h1>元素的所有段落。

组合选择符

你已经在本书中看到过组合选择符的例子。比如：你可以选取一个类选择符然后把它作为子孙选择符的一部分，像这样：

```
.blueberry p { color: purple; }
```

这里我们选择了blueberry类中一个元素子孙的所有段落。

这里有一种模式，你可用来构造一个复杂的选择符。我们一步一步来看看这种模式的用法：

(1) 首先，定义要选择元素的上下文，像这样：

```
div#greentea > blockquote
```

我们在这使用了一个子孙选择符<blockquote>，它的父选择符是<div>选择符，其id为“greentea”。

(2) 写出你要选择的元素：

```
div#greentea > blockquote p
```

上下文

元素

下一步，在想要选择的<blockquote>上下文后添加一个<p>元素，它必须是<blockquote>的后代，而<blockquote>是id为“greentea”的<div>选择符的孩子。

(3) 详细说明伪元素选择符或者伪类：

```
div#greentea > blockquote p:first-line { font-style: italic; }
```

上下文

元素

然后添加一个伪元素选择符，first-letter来选择段落的首行。

真是一个复杂的选择符！用这种方法就可以随意构造你的选择符。

#2 框架

HTML允许你把网页拆分为几个框架，每个框架都能在网页中显示。你可能已经注意到带有多框架的网页允许你访问第三方页面，同时保持标题和原导航点不受影响。如今框架的大部分已经被认为“过时”，因为它们造成了导航和可用性问题，并且W3C也不建议使用。然而，在一些地方你仍然会发现它们的有用之处。

使用**<frameset>**和**<frame>**元素在网页中创建一组框架：

创建了一系列框架作为三行，第一行占浏览器的30%，最后一行占浏览器的20%，中间那行占浏览器的50%。

你也可以定义列框架，或者兼有行和列的框架。

一个框架一个<frame>元素，每个框架指定了一个框架名和HTML源文件。

```
<frameset rows="30%, *, 20%">
   <frame name="header" src="header.html" />
   <frame name="content" src="content.html" />
   <frame name="footer" src="footer.html" />
</frameset>
```

以单个name为锚点，使用**<a>**元素在目标链接（target link）中指定框架名，像这样：

```
<a href="newpage.html" target="content">new content</a>
```

还有一个叫做**<iframe>**的关系元素，它能够被新版本的浏览器所支持。这个内联元素**<iframe>**允许你在页面的任何地方放置框架。下面是这个元素的用法：

为“newcontent.html”页面创建一个内联框架。

```
<iframe name="inlinecontent" src="newcontent.html"
        width="500" height="200" />
```

最后你需要知道，如果要使用框架,需在含有框架布局（frameset）的网页中使用DOCTYPE。框架布局的DOCTYPE被认为是一种过渡性文本,所以不能设置框架和规范。HTML 4.01用法：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
                      "http://www.w3.org/TR/html4/frameset.dtd">
```

XHTML 1.0 用法：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
              "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
```

#3 多媒体和Flash

浏览器可以播放声音、显示视频或者动画，比如网页中的Flash应用。HTML通过**<object>**元素支持这些类型的媒体。**<object>**负责在网页中嵌入外部内容（你的网页需要插件程序（plugin viewer），它知道如何显示内容）。

我们应该提到与**<object>**关系密切的堂兄——**<embed>**元素——绝不会在浏览器的战争中全军覆没，它们的使用要比它们本身复杂得多。如果你想在自己的网页中包含多媒体，我们鼓励你访问其媒体类型设计者的网站，并使用他们推荐给你的设置。虽然嵌入媒体比较复杂，不过不要被它吓跑——你已经有足够的HTML知识可以解决它，可以不用花太多时间就能在你的网页中添加声音、动画和电影。

一个简单的例子，使用一个**<object>**元素嵌入一个Quicktime电影：

这是object开始标签。正如你看到的，object中需要罗列出一系列的标签和属性，用以指定嵌入在网页中的纠错程序（corret viewer）。

```
<object classid="clsid:02BF25D5-8C17-4B23-BC80-D3488ABDDC6B"
        codebase="http://www.apple.com/qtactivex/qtplugin.cab"
        height="200"
        width="300">
   <param name="src" value="buckaroo.mov">
   <param name="autoplay" value="true">
   <param name="controller" value="true">
   <embed height="200"
          width="300"
          src="buckaroo.mov"
          pluginspage="http://www.apple.com/quicktime/download/"
          type="video/quicktime"
          controller="true"
          autoplay="true">Sorry your browser does not support this movie type</embed>
</object>
```

可以在object元素中嵌入一些元素来提供一组选择。假如浏览器不支持外部的<object>，可以试一试<embed>。

嵌入的<embed>元素用来支持旧版本的浏览器。

在你的网页中嵌入多媒体内容，能给你的用户留下更深刻的印象。不过指定**<object>**元素可能是一件麻烦的事情，所以你要查阅此类viewer产品文档信息中关于如何在你的网页中嵌入它们的viewers。

#4 创建网页的工具

现在你了解了XHTML和CSS，就可以站在一定高度来判断像Dreamweaver、GoLive、FrontPage这样的工具是否适合你了。所有这些应用工具试图提供一个所见即所得的（WYSIWYG）方式来创建网页。我们相信你已经掌握了足够的XHTML和CSS知识来达到这个目标，同时，有时候还可以达到简洁的目的，但是另一方面，即使你自己已经写了很多的XHTML这些工具还提供了非常便利的特征。

- 一个“代码”窗口，用来进入具有语法检测功能的XHTML和CSS，它能够在你输入的时候检查出常犯的错误，还能够提供一些常用的名字和属性。
- 预览和出版的功能，允许你在网页存活于Web之前测试它们。
- 网站管理员，允许组织你的网站，还能使你的本地语法修改（local changes in synch）与你在浏览器上的网站保持联系。注意：这通常会帮你关照好所有的FTP任务。

这些工具不包括下面这些方面：

- 通常，这些工具偏离了标准支持,所以为了保持你XHTML和CSS的时效性——你需要自己书写XHTML。
- 通常，这些工具都没有严格遵循标准，并且可能允许你使用XHTML和CSS做很不规范的编写，所以不要忘记校验（同时有些工具能够帮助你校验）。

谨记：你可以使用简单编译器连同这些更复杂工具的组合；一个解决方案并不能满足你所有的需求。所以，当这个创建网页的工具有意义时，你再去使用它。

一些可以考虑的工具：

- Macromedia Dreamweaver
- Adobe GoLive
- Microsoft FrontPage
- GNU Emacs(open source)

#5 客户端脚本

HTML网页不是被动的文档；通常有些内容是可执行的。可执行内容会对网页产生影响。你可以通过编写程序或者脚本的方式创建可执行内容。这里有许多配合浏览器工作的脚本语言，其中JavaScript脚本语言起主导作用。这是一个把可执行内容放到你网页中的小小尝试。

```
<script type="text/javascript">
   function validBid(form) {
      if (form.bid.value >0) return true;
      else return false;
   }
</script>
```

一个新的HTML元素，即<script>，它允许你在HTML中编写代码。注意，我们还设置了JavaScript的类型。

这里有一点JavaScript的脚本，它用来检查用户的出价是否是0美元还是低于零美元。

然后，在XHTML中，你能够创建一个表单，使用这个脚本来检查在表单提交之前用户的出价。如果出价大于0，表单就能够提交。

```
<form onsubmit="return validBid(this);" method="post" action="contest.php">
```

这是表单中一个新的属性，叫做onsubmit，当提交按钮按下时调用脚本。

脚本还可以做些什么？

看过上面的内容后知道，表单输入校验是一种普通的、有用的工作，这个工作通常由JavaScript完成（这个例子之外有其他校验类型你可以了解）。不过，这只是你使用JavaScript的开始而已。JavaScript 实际上访问的是整个文档元素树（和第3章一样的元素树），通过编程可以改变文档树中的值和元素。这是什么意思呢？意思就是说你可以基于用户行为来编写一个脚本以改变网页的不同方面。使用JavaScript可以做这些事：

- 创建一个交互式的游戏，比如填字游戏。
- 当用户把鼠标移到图片上时，可以动态地改变图片。
- 在用户的浏览器中设置本地信息，那么你就能够在他们下次访问时记住这些信息。
- 让用户选择网页的不同样式。
- 从一组格言中随机显示一个格言。
- 显示圣诞节前的购物天数。

#6 服务器端脚本

许多网页都不是由手工创建，而是由运行在服务器上的Web应用程序创建的。例如：一个在线的订购系统，服务器在一步步的订购过程中创建网页；或者，一个在线论坛，服务器针对存储在数据库某处的论坛信息创建网页。我们通过一个Web应用程序处理你在第14章里为Starbuzz Bean Machine创建的表单。

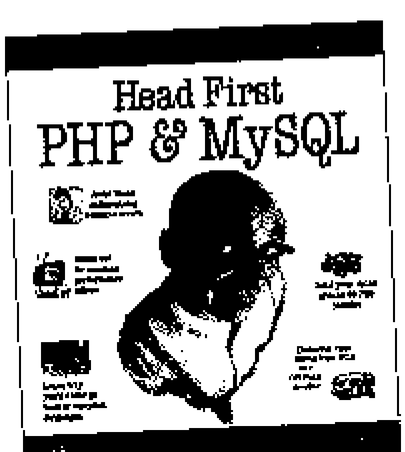
许多服务器代理商允许你通过编写服务器端脚本和程序的方式创建自己的Web应用程序。下面是服务器端脚本允许你做的几件事情：

- 建立一个在线商店包括商品、购物卡及一个订购系统。
- 个性化用户喜爱的网页。
- 发布最新新闻、事件、消息。
- 允许用户直接搜索站点。
- 允许你的用户帮助你构建网站内容。

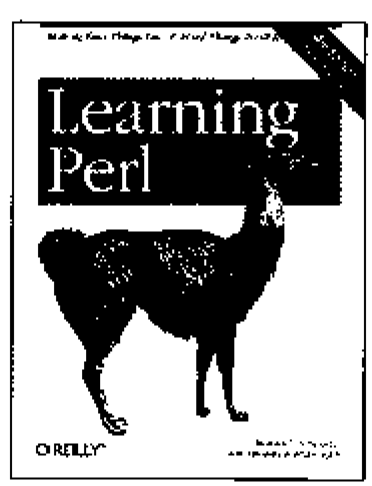
要创建一个Web应用，你需要懂得一些服务器端脚本或程序语言。在网络的发展过程中关于什么语言最好有很多的争议，问的人不同，得到的回答也不同。事实上，网络语言就好像汽车：你能够开任何一辆车从Yugo到达Hummer，每辆车都不同，都有它自身的优点和缺点（费用，使用的便捷性，是否经济等）。

Web语言正不断地被开发出来：Python, Perl , Ruby on Rail, Java Sever Pages（JSP）都是常用的语言。如果你刚开始编程，那么用PHP语言编程是最容易的。网络上有数百万计的PHP-driven 网页，学好它之后你就能进入好公司。另一方面，如果你有编程的经验，就不妨试试JSP。假如你对Microsoft技术更感兴趣，那么你可能会把VB.net和ASP.net作为服务器端的解决方案。

这里有一些书你可能会用得着：



来自2006年





#7 转到搜索引擎

许多用户通过搜索引擎（比如Google, Yahoo）找到你的网站。在有些情形下，你不希望你的网站出现在搜索引擎的搜索列表中，那么你可以使用XHTML来请求不被列人。不过，在其他情形下，你会尽你所能去调整站点，当搜索特定词汇时让你的网页尽可能地出现在前面。这里有些常用的技巧可以改进你的网页搜索结果。不过谨记，每个搜索引擎的工作原理不同，而且在决定排名顺序时考虑的因素也不同。

提高你的排序

搜索引擎通过网页中的组合单词和词组进行搜索排序。所以，为了提高网页排序并告知搜索引擎网页相关内容，需要在**<head>**开始处添加两个**<meta>**标签：一个列举关键字，另外一个提供内容描述。关键字是一两个描述网站的单词，比如“咖啡” 或者“旅行日志”。

```
<meta name="description" content="This would be your description of what
is on your page. Your most important keyword phrases should appear in this
description." />

<meta name="keywords" content="keyword phrase 1, keyword phrase 2, keyword
phrase 3, etc." />
```

与文本中其他内容相比，很多搜索引擎更看重标题及**alt**属性和**title**属性中的文字，所以确保在这些元素、属性中书写简明有意义的文本。

最后，许多搜索引擎会考虑从其他网站到你网站的链接数目；链接到你网站的站点越多，你的网站就越重要。所以，你所能做的就是鼓励别人链接到你的站点来提高你的搜索排序。

如何阻止我的网站列入搜索列表？

你可以请求搜索引擎忽略你的网页，不过不能保证所有的引擎都会这么做。真正阻止他人找到你的网站的唯一方法就是将你的网站设置为私有（和你的服务器代理商商量商量）。不过，你的网站通常同大多数主要的搜索引擎合作，假如你想请求你的网站不被搜索引擎列出，只需在你的XHTM表头中插入一个**<meta>**标签就行了，像这样：

```
<meta name="robots" content="noindex,nofollow" />
```

这个<meta>标签告诉搜索引擎忽略这一页及该页上链接的其他页面。

#8 关于打印的样式表

正如你在第10章中所看到的那样，你可以使用<link>元素的media属性指定一个可选媒体类型。如果你在一个样式表media属性中指定了一个“print”值，那么在打印页面时就会用到这个样式表。用<link>元素这样实现：

```
<link rel="stylesheet" type="text/css" media="print" href="forprint.css" />
```

link元素的media属性告诉浏览器打印网页时应该使用此样式表。

这链接到打印样式表。当监视器监测到你的网页时，不使用样式表；当用浏览器打印网页时，才会使用样式表。

接着，正如你已经看到的，当用户访问你的网页并且选择浏览器打印功能时，在打印网页之前，浏览器会应用“forprint.css”样式表，这允许样式化你的网页以便更适合打印。在调用打印的样式时，有几点需要你记住：

- 把打印区的文本背景颜色设置为白色，这可以增强被打印网页的可读性。
- 在你的样式表中用点比用像素、百分数或者em来定义字体大小要好得多。点是特别为打印文本设计的，对大多数字体来说，典型的点的大小是12pt。
- Sans-serif字体在显示屏上易读，而serif字体在打印网页上易读。你可以使用你的打印样式表来改变你的font-family。
- 如果你的导航菜单、工具条或者网页的主要内容周围的其他内容，对主要内容而言无关紧要，那么你可以为打印版本的网页隐藏这些元素。通过把这些元素的显示性质设置为“none”来完成隐藏。
- 如果你已经在网页中将元素定位，你可能考虑到移除定位属性，那样网页可以用综合的方式打印内容，这对内容的读取有很大意义。
- 如果你已经在网页中设置了元素的宽度，你可能会用margins或者其他方法把这些宽度改为灵活的宽度。如果你的网页有特定的宽度，那么它可能并不适合打印页面。

制作好的打印样式表的关键是着眼于网页的主要内容，并且确保这些内容打印清晰，适合打印页面，并且易读。要想知道你的网页在打印出来时看起来是否好看，最好的方法就是用打印页面来测试你的打印样式表。

#9 适合移动设备的网页

想让你的网页能够在移动设备上使用吗，例如在手机和PDA（个人数码助手）上？随着移动设备的不断发展，它们对XHTML和CSS的支持也因种类繁多而不同：有些支持CSS，有些不支持CSS；有些能很好地显示XHTML，有些则会出现混乱。你能做的就是发现潜在问题以求日后能得到更好的支持。

首先，记住你可以使用<link>元素的media属性书写一个“handheld”特殊样式表。

```
<link rel="stylesheet" type="text/css" media="handheld" href="formobile.css" />
```

使用值为“handheld”的media属性为移动设备创建一个样式表。

不幸的是，支持“handheld”样式表的媒体类型始终是有限的。所以就算你的网页中链接了一个“handheld”样式表也不代表手机上的浏览器能使用它。所以，你要留意常用的设计技术使得网页在电脑监视器和小规模设备中都能很好地显示：

- 记住许多移动服务器按照数据流量收费。这是我们编写简短、正确的XHTML并使用CSS来样式化你网页的最佳理由。
- 保持导航简单易见。这意味着你要是用文本链接并避免奇特的脚本效果。比如要求使用鼠标和键盘。
- 尽可能缩小网页比例。如果你有handheld样式表，用它来尽可能减小字体尺寸、边界和边框。
- 注意你的多栏布局常常在小设备上被忽略，所以留意你的XHTML中元素的排列。
- 有些移动设备不支持框架和弹出窗口，所以避免使用它们。
- 最后，最好的办法就是经常测试你的网页，这样你才能知道它们在小设备上的真实表现。

在当前支持有限的情况下，如果你有为“handheld”媒体类型书写可选择样式表的习惯，你会为将来的学习打好基础，并能为它们提供更多的支持。

#10 blog博客

网络日志——或者叫博客比较通俗——如同个人网页，除了是以旅行形式书写，例如Tony的网络旅行日志，许多创建blog的人使用Web服务器来控制blog的全部细节。服务器同样提供定制好的模板，这允许你从各种样式中选择你喜欢的外观。它们提供不同的背景颜色、字体样式，甚至可以使用背景图片，它们同样允许你自定义blog模板并创建自己特有的blog外观，所使用的是，你已经猜到了——XHTML和CSS。

这是一个XHTML和CSS片断，它来自一个流行的在线blog网站Blogger.com上的blog模板。正如你所见，它们使用和你在本书中学到的一样的元素和属性，它们甚至是最新标准：使用的是XHTML 1.0 Strict，所以学过如何严格编写XHTML是一件多么棒的事情，对不对？让我们再详细了解一下……

这个blog服务器使用XHTML 1.0 Strict，所以这里的DOCTYPE和<html>属性都是你之前见过的。

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">

<head>

  <title><$BlogPageTitle$></title>

  <$BlogMetaData$>

  <style type="text/css">
    body {
      margin: 0px;
      padding: 0px;
      text-align: center;
      color: #554;
      background: #689C54 url(http://www.blogblog.com/dots_dark/bg_minidots.gif) top center repeat;
      font-family: "Lucida Grande", lucida, helvetica, sans-serif;
      font-size: small;
    }
    .
    .
    .
  </style>
</head>
<body>
  .
  .
  .
</body>
</html>
```

这个<$...$>是模板变量，当你创建blog和添加新POST时，它们会用你的blog名和其他内容来填充，当它需要正确显示你的blog内容时，你应该保留这些变量。

这是样式表的顶部，它确定你blog的外观。这个模板消除了主体的边界和补白，给文字确定了一个默认颜色，给页面背景配置了一张图片，并设定了字体属性。

这里有许多样式规则。每个样式规则分别控制blog的显示如字体、标题、颜色……。也就是说，现在这些和你以前见过的样式一样。

这些XHTML包括了blog中所需的各个部分：标题、内容、日期等。每个内容区域都会有<$...$>变量，用来插入从你的post中获取的内容。

索引

符号

&amp；114
&gt；114
&lt；114
.. 64，65
<a> 48，152
<address> 116
<blockquote> 90
<body> 23

 98
<code> 116
<dd> 108
<del> 377
<dl> 108
<dt> 108
<fieldset> 632
<form> 594，596
<h1> 22，25，82
<h2> 22，82
<head> 23
<hr> 116
<html> 23
<img> 47，82，173
<input> 598
<ins> 377
<label> 632
<legend> 632
 104
<meta> 240
 105
<option> 601
<p> 22，82
<pre> 116
<q> 86
<select> 601
<strong> 116
<style> 29
<table> 551，554
<td> 554
<textarea> 600
<th> 554
<title> 23
<tr> 554
 105
404 错误 137

A

绝对路径 136，138～139
可访问性 149，159，176，251，255，270，355，479，558，632
Adobe GoLive 643
Adobe Photoshop元素 184
锚 155
抗锯齿（anti-aliasing） 213
属性 158. 参见元素
  行为 596，605
  Alt属性 176
  colspan属性（即单元格占几列，用于合并单元格） 571
  content属性 240
  height属性 177
  href属性 48～49
    访问 53
  http-equiv属性 240
  id属性 152，416～419
  media属性 649
  method属性 594，605

GET与POST 620～621
onsubmit属性 645
rel属性 305
rowspan属性（即单元格占几行，用于合并单元格） 569
target属性 158～159
缺陷 159
title属性 149
width属性 177

B

背景图像 404～407
Bean Machine 602
幕后 48，167
成为浏览器 112，327，490，556，618，634
Blogger.com 650
博客 650
边框 参见属性，边框
箱模型 391～395
边界 396，401～403
填充 396，401～403
宽度 444～445
智能 9，46，65，103，105，158，225，237，267，290，293，398，415，435，439，443，462，467，481，551，575，576，612，623
浏览器
Firefox 16
Internet Explorer 16
歌剧 16
Safari字体 16
浏览器之战 226
Bullet Points 36，69，117，161，214，261，331，379，424，482，542，581
Burma Shave 108

C

Caffeine Buzz 144
级联样式表 32，285
XHTML 289
级联 473～480
简要描述 322
类 318，320
颜色 366～368
十六进制 369～371
扩展的 303～305
流程 489
边界 493
浮动元素 495～496
内联元素 491
清除属性 511
宽度 495～496
用于打印 423
用于各种媒介 422～423
继承 311～312
覆盖 314
语言 289
倍数 420～421
属性 参见属性
规则 289，296
语法 289
选择器 294，297，300
属性 640
子 454
类 318
组合复杂性 640
后代 453
元素 297
组 297
id属性 417
倍数 322
伪类 469
伪元素640
同属 640
速记 458～461
规范 476
确认 329
字符编码 239
ISO-8859-1 240
字符实体 114～115

结束标记 25
颜色 364
    颜色图 373
    十六进制码 369
        快捷方式 373
    红，绿，蓝 364～365
    Web颜色 372
Colspan属性 571
注释
    HTML 6
遵从 230
填字游戏 37，72，118，160，215，262，283，332，380，425，543，582
CSS 参见级联样式表
对话 224，233，248，266，432，451，463

D

默认页 141
设计
    Jello 518
    流体和固体 517
    略图 81
    草图 80，81
    三列 541
    两列 503
目标锚 152～155
DOCTYPE 229，245，246. 参见文档类型定义
文档类型定义 229，245
    增加 231
域名 128～129
Dreamweaver 16，134，643
DTD 229

E

Earl的汽车 138
元素 25
    A元素 47～49，144
        最佳实践 150
        目标锚 152～155
        样式 468
        标题 149
    地址 116
    属性 29
        定义 51～53
        语法 51
    块 94
    块引用 91～94
    body元素 23
    br元素 98～101
    标题 557
    选项框 607
    结束标记 25～26
    代码 116
    定义 25
    div元素 432
        类 437
        用于逻辑划分 434
        标注 434
        嵌套 436
        排序 513
        定义样式 440
    dl 108
    dt 108
    em 116
    embed 643
    empty 100～101
    fieldset 632
    表单
        活动属性 596，604
        如何工作 592～593
        输入元素 598～601
        Method属性
        post属性 605
    method属性 596，605
        post属性 596
    定义样式 630
h1 23
h2 23

head 23
hr 116
href 144
html 23
img 166～169，173
<a> 元素 202
alt属性 176
URL 174
高度属性 177
如何工作 167～169
加载175
尺寸182
宽度属性 177
内联 94
输入 598～601，605，606
选项框 599，615
文件 633
单选 599，613
submit属性 598，607
text属性 598
label属性 632
li元素 104
标志 579
定义样式 578～580
链接 305，648
匹配标记 25～26
元素
meta 647
嵌套 109
对象 643
ol 105
起始标记 25～26
选项 601，607，633
p 23
口令 633
pre 116
q 86
引用 参见元素，q；元素，块引用
根元素 110
选择 601，610～611
使用多重选择 633
span 463～465
strong 116
style 29，291
小结 557
表 552
压缩边框 564
dissected 554
嵌套 573～574
填充和边框 562
行与列的跨度 569～571
定义样式 560，576
td属性 552
正文区域 599，615
正文输入t 607
th属性 552
tr属性 552
ul属性 116
饮料 430
Emacs 643
空元素 100～101
实体 114～115
可扩展的HTML 227，266
比较HTML 268～269
DOCTYPE 274
空元素 275
从HTML移植 272
命名空间274
使用 270～271
确认 277
HTML 282
XML 267

F

文件传输协议132～134
Firefox 16
围炉夜话
CSS和XHTML语言比较 324
Float和Absolute讨论更适合布局 530
HTML和CSS聊内容和样式 34

HTML和XHTML争赢支持率 280
内联元素和块元素公布他们的不同 96
JPEG和GIF比较它们的图像 170
表格和CSS争论如何布局表单 624
5分钟之谜
蛮力对样式的案件 302，308
相对与绝对之争 148，156
双胞胎奇案 89，93
flash 643
文件夹
组织 56～57
父 63
相对路径 59～64
子文件夹 60
树 56
字体 358
尺寸 参见属性，font-size
样式 参见属性，font-style
字符. 参见properties，font-weight
font-family 344～347，350
处理浏览器 351
font-size 352，354～355，358
尺寸 352～354
font-style 361
font-weight 359
字符
Serif与sans-serif 350
表单. 参见元素，表单
框架 642
文档类型 642
FrontPage 16
FTP. 参见文件传输协议

G

GET 620
GIF 206
JPEG 170～171，175，192
GoLive 16，134，643

H

Head First学习原则 xxviii
Head First休闲室 4，44，231，237，246，268，291，386，430，488，497
主机代理商 126～127
HTML. 参见超文本标记语言
严格的HTML入门 4.01 252～254
head与body 23
http 参见超文本传输协议
超文本标记协议 4.01 227
注释 6
遵从 230
定义 6
过期 259
元素. 参见元素
历史记录 226
标准 230
严格 243～246，252～255
入门 4.01 253
适当嵌套 253～255
过渡 230
确认 234
字符编码 239
XHTML 282
超文本传输协议 136～137

I

图像. 参见img元素与<a> 元素 203
抗锯齿（anti-aliasing） 213
GIF 169
JPEG 169
加载Web页面 175
matte－color 209
品质 190
重新定义尺寸 186～189
缩略图 195～198
透明 207
URL 175
继承 311

Internet Explorer 16
本周访问
用类总是恰当的吗？ 414
href属性的告白 53
开始了解伪类 470
使用target被认为很糟糕? 159
为什么你会关心你使用的是哪个版本？ 228
ISO-8859-1 240

J

Java 646
JavaScript 645
Java Server Pages 646
JPEG 169，189
品质 190
GIF 170～171，175，192

L

换行 94～95
链接 参见元素，a
列表 103～105，108. 参见元素、li、ul、ol
定义样式 577～580
逻辑划分. 参见logical section
逻辑部分 433～437. 参见元素，div
Head First休闲室

M

Macintosh
创建文本文件 12
Dock 12
TextEdit 12
标记
分解元素 25
标记帖 21，298，603，635
匹配标记 25，26，111
粗的 207
Photoshop元素的选项 208
设置颜色 209
Microsoft FrontPage 643
多媒体 643
myPod 178
MySQL 646

N

嵌套列表 108～110
嵌套 109
记事本 14

O

打开页面，浏览器 19
开放的tag 25
歌剧 16

P

页面排序 647
父 63，65
Perl 646
Photoshop元素， 参见Adobe Photoshop元素
PHP 141，646
像素 182
纯文本 12
PNG 175
端口 147
配置
绝对 520～522，532
浮动 530～531
固定 535～537
相对 539～540
Z轴 521～522
POST 620
属性 289～290
background-color 289，330，367～368
background-image 330，404～407
边框 289，396，410～411
border-bottom 295，378
bottom 532
清除 511
颜色 292，343

浮动 488，495～496
font-family 342
处理浏览器 351
font-size 342～345
keywords 354～355
percent与em 358
像素 358
font-style 330，361
样式 361
font-weight 330，343，359～360
字符集 344～347
left 330，521
letter-spacing 330
line-height 330，389～390
list-style 330
边界 330，396
填充 396
位置 520. 参见位置
right 520
速记 458～461
text-align 330
图像 449
text-decoration 343，375，377
top 330，520，521
属性 289
协议136
文件 147
伪类469，469～471
Python 646

Q

Quicktime 643
奇怪的方式 228～229

R

相对路径 59
相对路径 147
重定义图像大小 186～188
根元素 110
根目录 131
Rowspan属性 569
Ruby语言 646
Rails架构的Ruby语言 646

S

考察队 16
sans-serif字体 309
保存你的成果 18
脚本 645
搜索引擎 647
Segway 78
选择器 294，297，300，318，322，453，469，640～641.参见级联样式表
Serif字体 309
服务器端脚本 646
Servlets 646
Sharpen your pencil 7，10，79，106，180，301，310，319，356，359，398，446，450，466，472，508，533，563，565
缩写 458～461
span 462～467. 参见元素，span
具体度 322，475～480. 参见级联样式表
Starbuzz咖啡 9，126，144，499，602
结构 5，87
样式表 参见级联样式表

T

表 参见元素，table
表现 555，624
标记 5
meta 240
解剖标记 25
目标 158
TextEdit 12
参数选择 13
保存18
Internet 2
There are no dumb questions 6，16，57，92，108，115，129，131，134，155，158，175，177，190，

203，213，230，240，251，255，276，277，290，297，315，329，350，358，362，373，377，396，419，421，437，446，454，461，466，469，479，494，512，522，555，563，571，580，607，621
三列布局 541
缩略图 196
Tony的日志 79，278，348，550
透明 205. 参见GIF

U

统一资源定位器（URL） 136，147
  绝对路径 136
  默认页面 141
  http 137
  端口 147
  协议 136，147
URL 参见统一资源定位器

V

确认 233～236，238，242，247，250. 参见超文本标记语言，级联样式表，可扩展的HTML

W

W3C 233，234
注意! 86，355，412，532，535
Web浏览器 3
博客日志 650
网页 2
Web服务器 3
严格HTML 4.01入门 252～254
连连看 32，113，570
Windows
  创建文本文件 14
  文件扩展名 15
  记事本 14

X

XHTML 227，268～269. 参见可扩展的HTML
  检查列表 272
  使用 270
  HTML 280～282
XML 267

Z

Z轴 521～522，529. 参见位置